

**Earth Science Data Information System (ESDIS)**

**Data Gathering and Report System (EDGRS)**

**White Paper**

**Describing the EDGRS Analysis and Design Approach**

written by

**Computer Sciences Corporation**

**September 2004**



## Abstract

TBS



# Table of Contents

<b>1</b>	<b>INTRODUCTION .....</b>	<b>1-1</b>
<b>2</b>	<b>BACKGROUND .....</b>	<b>2-1</b>
2.1	Brief history of SCRS.....	2-1
2.2	Performance Measurement of ECS .....	2-2
<b>3</b>	<b>OPERATIONS CONCEPT FOR EDGRS .....</b>	<b>3-1</b>
3.1	Data Collection at DAACs.....	3-1
3.2	Central database.....	3-2
3.3	The EDGRS Database Environment .....	3-3
3.4	Reporting .....	3-4
<b>4</b>	<b>EDGRS REPORTS AND GRAPHS.....</b>	<b>4-1</b>
4.1	Annual Report .....	4-1
4.2	Ad-hoc and Special Reports .....	4-1
<b>5</b>	<b>DATABASE DESIGN .....</b>	<b>5-1</b>
5.1	Database Segmentation.....	5-1
5.1.1	Description .....	5-1
5.1.2	Why Use Segmentation? .....	5-1
5.1.3	Limitations .....	5-2
5.1.4	Offline vs Online Segments.....	5-2
5.1.5	Subsystems and Tables Affected .....	5-3
5.2	Summary Tables.....	5-4
5.2.1	Description and Justification .....	5-4
5.2.2	Last 30 Days Method .....	5-4
5.2.3	Correction for Null ordered Values .....	5-4
5.2.4	Correction for non-null orderids spanning multiple days .....	5-5
5.2.5	Difference between time fields used for segmentation and time fields used for summary reports 5-5	
5.3	EDGRS Ingest Subsystem .....	5-6
5.4	EDGRS Archive Subsystem .....	5-15
5.5	EDGRS Distribution Subsystem .....	5-20
5.5.1	Overview .....	5-20
5.5.2	Building the EDGRS dist Table .....	5-33
5.5.3	Building the EDGRS SummaryTables Based on the dist Table.....	5-38
5.5.4	Building the EDGRS SCRS Tables.....	5-59
5.5.5	Building the EDGRS SCRS SummaryTables .....	5-78
5.5.6	Building the EDGRS User Characterization Tables.....	5-82

5.5.7	Building the EDGRS ecac Tables .....	5-90
5.5.8	Building the EDGRS dsdd Tables.....	5-94
5.5.9	Ingesting the EDGRS Daily gsfcv0 Data .....	5-99
5.5.10	Building the OM Tables(TBS) .....	5-105
5.5.11	Building Other Supporting Tables (TBS).....	5-105
<b>6</b>	<b>PROCESS CONTROL (TBS)</b> .....	<b>6-1</b>
<b>APPENDIX A</b>	<b>EDGRS REPORT DEFINITION TABLE</b> .....	<b>A-1</b>
<b>APPENDIX B</b>	<b>DETAILED TABLE DESIGN</b> .....	<b>B-1</b>
<b>APPENDIX C</b>	<b>PROCESSING CONTROL STRUCTURES</b> .....	<b>C-1</b>
<b>APPENDIX D</b>	<b>DETAILED DESCRIPTION OF CORRECTION FOR ORDERS SPANNING DAYS</b> .....	<b>D-1</b>

## List of Figures

Figure 3-1 General Data Flow .....	3-1
Figure 3-2 Database Ingest .....	3-2
Figure 3-3 Simplified Database Layout .....	3-4
Figure 3-4 EDGRS Report Processing.....	3-5
Figure 5-1 Ingest Data Flow Diagram (DFD).....	5-7
Figure 5-2 Ingest – Last 30 Days DFD .....	5-10
Figure 5-3 Archive – Full DFD.....	5-16
Figure 5-4 EDGRS Distribution Subsystem – Base Table Overview.....	5-24
Figure 5-5 EDGRS Distribution Subsystem Summary Table Overview – Order/Subscription Data..	5-30
Figure 5-6 EDGRS Distribution Subsystem Summary Table Overview – Log Data .....	5-32
Figure 5-7 Distribution – Dist Ingest DFD .....	5-35
Figure 5-8 Dist Report Generation – from Archive DFD .....	5-39
Figure 5-9 Dist Report Generation – to Users DFD.....	5-42
Figure 5-10 Dist Report Generation – Subscriptions DFD .....	5-45
Figure 5-11 Dist – Last 30 Days DFD.....	5-47
Figure 5-12 Dist Report Generation – Order Status DFD .....	5-50
Figure 5-13 Dist Report Generation – Orders Spanning Days DFD.....	5-52
Figure 5-14 Dist Report Generation – Order Counts DFD.....	5-54
Figure 5-15 SCRS Ingest – ftp DFD.....	5-60
Figure 5-16 SCRS Ingest – wdd DFD .....	5-65
Figure 5-17 SCRS Ingest – www DFD .....	5-69
Figure 5-18 SCRS Ingest – inq DFD .....	5-74
Figure 5-19 SCRS Report Generation DFD .....	5-79
Figure 5-20 ecac Users – Ingest DFD .....	5-83
Figure 5-21 usrprofile Users – Ingest DFD.....	5-86
Figure 5-22 ecac Orders – Ingest DFD .....	5-91
Figure 5-23 dsdd Ingest DFD.....	5-95
Figure 5-24 gsfcv0 Ingest DFD .....	5-100

## List of Tables

Table 5-1 Tables Affected by Distinguishing Between Online and Offline Segments .....	5-2
Table 5-2 Base Table Segmentation in EDGRS .....	5-3
Table 5-3 Summary Tables Using the Last30days Method .....	5-4
Table 5-4 Summary Tables Affected by Null Orderids.....	5-5
Table 5-5 Summary Reports Affected by Orders Spanning Multiple Days.....	5-5
Table 5-6 Different Time Fields Used for Segmentation and Report Generation .....	5-5
Table 5-7 Ingest DFD Key .....	5-8
Table 5-8 Last 30 Days DFD Key .....	5-11
Table 5-9 Ingest Subsystem Table Definitions .....	5-13
Table 5-10 Ingest Subsystem Index Definitions .....	5-14
Table 5-11 Full DFD Key .....	5-17
Table 5-12 Archive Subsystem Table Definitions .....	5-18
Table 5-13 Archive Subsystem Index Definitions .....	5-20
Table 5-14 Mapping of Distribution Methods to Tables used in the Data Import Process.....	5-21
Table 5-15 Description of Tables in Dist Subsystem .....	5-25
Table 5-16 Description of dfa and dtu Summary Tables in Distribution Subsystem .....	5-31
Table 5-17 Description of SCRS Summary Tables in Distribution Subsystem.....	5-32
Table 5-18 Distribution – Dist Ingest DFD Key .....	5-36
Table 5-19 Definitions for Tables Used to Ingest Data into dist from FoxPro .....	5-37
Table 5-20 Index Definitions for Tables Used to Ingest Data into dist .....	5-38
Table 5-21 Dist Report Generation – from Archive DFD Key .....	5-40
Table 5-22 Dist Report Generation – to Users DFD Key .....	5-43
Table 5-23 Dist Report Generation – Subscriptions DFD Key .....	5-46
Table 5-24 Dist – Last 30 Days DFD Key .....	5-48
Table 5-25 Dist Report Generation – Order Status DFD Key .....	5-51
Table 5-26 Dist Report Generation – Orders Spanning Days DFD Key .....	5-53
Table 5-27 Dist Report Generation – Order Counts DFD Key .....	5-55
Table 5-28 Definitions for Tables Used to Build Summary Tables for dist.....	5-56
Table 5-29 Index Definitions for Tables Used to Ingest Data into dist .....	5-58
Table 5-30 SCRS Ingest – ftp DFD Key.....	5-61
Table 5-31 Definitions for Tables Used to Build the SCRS_ftp Table .....	5-63
Table 5-32 Index Definitions for Tables Used to Build the SCRS_ftp Table .....	5-64
Table 5-33 SCRS Ingest – wdd DFD Key .....	5-66
Table 5-34 Definitions for Tables Used to Build the SCRS_wdd Table.....	5-67
Table 5-35 Index Definitions for Tables Used to Build the SCRS_wdd Table .....	5-68
Table 5-36 SCRS Ingest – www DFD Key .....	5-70
Table 5-37 Definitions for Tables Used to Build the SCRS.www Table .....	5-71
Table 5-38 Index Definitions for Tables Used to Build the SCRS.www Table.....	5-72
Table 5-39 SCRS Ingest – inq DFD Key .....	5-75
Table 5-40 Definitions for Tables Used to Build the SCRS_inq Table .....	5-76
Table 5-41 Index Definitions for Tables Used to Build the SCRS_inq Table.....	5-77
Table 5-42 SCRS Report Generation DFD Key.....	5-80
Table 5-43 Definitions forTables Used to Build SCRS Summary Tables .....	5-81
Table 5-44 Index Definitions for Tables Used to Build SCRS Summary Tables .....	5-82
Table 5-45 ecac Users – Ingest DFD Key.....	5-84
Table 5-46 usrprofile Users – Ingest DFD Key .....	5-87
Table 5-47 Definitions for Tables Used to Build User Characterization Tables .....	5-88
Table 5-48 Index Definitions for Tables Used to Build User Characterization Tables .....	5-90
Table 5-49 User Characterization Table Definitions.....	5-90
Table 5-50 ecac Orders – Ingest DFD Key .....	5-92
Table 5-51 Definitions for Tables Used to Build ecac Tables.....	5-92
Table 5-52 Index Definitions for Tables Used to Build ecac Tables.....	5-93
Table 5-53 ecac Orders – Ingest DFD Key .....	5-96
Table 5-54 Definitions for Tables Used to Build dsdd Tables.....	5-97
Table 5-55 Index Definitions for Tables Used to Build dsdd Tables .....	5-98
Table 5-56 gsfcv0 Ingest DFD Key .....	5-101

<b>Table 5-57 Definitions for Tables Used to Ingest the gsfcv0 Data .....</b>	<b>5-103</b>
<b>Table 5-58 Index Definitions for Tables Used to Ingest the gsfcv0 Data .....</b>	<b>5-104</b>

## **1 Introduction**

The National Aeronautics and Space Administration (NASA) established the Distributed Active Archive Centers (DAACs) as data repositories and distribution points for various types of data collected from the satellites in the Earth Observing System (EOS) satellite series. The Earth Science Data Information System (ESDIS) Project is responsible for the development of the data gathering, data management, and distribution mechanisms for the Earth Observing System Data and Information System (EOSDIS). More specifically, the ESDIS Science Operations Office (SOO) manages the DAACs for this data collection and distribution purpose.

To effectively assist in the management of the DAACs, the SOO needs to keep statistics and generate reports that portray the effectiveness of the DAAC function. The implementation of this concept has developed in two phases. The first phase involved the development of the Version 0 (V0) systems which represent the initial data ordering environment, and the Statistics Collecting and Reporting System (SCRS), which is primarily a data collection and reporting mechanism that was put in place quickly while the development of the formal EOSDIS Core System (ECS) proceeded. The ECS is a system that resides at specific DAACs to support the ESDIS data processing and distribution.

The ESDIS Data Gathering and Report System (EDGRS) is intended to be the metrics collection agent in the EOSDIS environment. The intended audience of the EDGRS metrics is NASA management, with the intent to support SOO operations in the ECS era.

## 2 Background

### 2.1 Brief history of SCRS

The SCRS was established to provide information on the amount of data being delivered from the pre-ECS (V0) EOSDIS Distributed Active Archive Centers (DAACs) and the number of requests for data and information being received and processed. The SCRS received and processed data from March 1995 through September 2002. The SCRS reports were the primary means of reporting the overall DAAC statistics during this time period. SCRS measured DAAC usage and gives an indication of DAAC performance. It indirectly measured user satisfaction, and provided some user characterization. The non-ECS data is now merged with the ECS data in the EDGRS environment.

A primary objective of EOSDIS is to efficiently and responsively provide data and information services to users. Statistics on the services the users receive are the measure of how successful EOSDIS is at meeting this objective. The SCRS was established to provide information on the amount of data being delivered from the EOSDIS DAACs and the number of requests for data and information being received and processed. The statistics give information on how well that is being done, as well as providing the information needed to evaluate the V0 operations and to provide information for predictions of the future.

Each of the DAACs provides information on the requests for products received and processed, on the accesses to the DAAC local information management system (IMS), FTP sites, or World Wide Web (WWW or Web) pages, and on the number of inquiries received. Additional data are provided by the V0 System IMS, by the Global Change Master Directory (GCMD), and by the WWW V0 Gateway. These are input to databases that are the source for monthly SCRS reports as well as ad-hoc reports and analyses. Summary plots of the statistics are prepared as well, over fiscal and/or calendar year time spans.

The User Services Working Group (USWG) began to collect metrics on system use in response to requests from NASA, as well as needs from within the project. The initial attempts were to collect the data in the same format as was used for the reports. The first attempts in 1993 and 1994 led to requests for more data. A form was created for the User Support Office (USO) personnel at the DAACs to use, which became very large, with 300+ items for each DAAC to collect. It became apparent that differences in processing and in interpretation of the input to the matrices were invalidating the results. Also, it was much less efficient for each of the DAACs to perform the same operations separately to generate the reporting matrices. In addition, the content of the matrices limited the reports that could be generated from the data.

The spreadsheet was remodeled to be the “Fab Four”: the number of unique users, the number of accesses, volume distributed, and the number of new orders received. These values were provided on a monthly basis. The Fab Four were used for 1994 and much of 1995.

The Fab Four were still difficult to collect. They served a good purpose, but were only a temporary measure until a method that was less dependent on hand collection could be implemented.

As a result of an internal workshop to analyze what information was needed for DAAC management, it was decided that the only way to get consistent statistics was to collect raw data and process using a common procedure. In November-December, 1994, it was proposed to the DAACs that data be generated from each DAAC at the various steps of order fulfillment, sent to ESDIS, and that ESDIS take the job of doing the counting. Numerous discussions were held, which resulted in an ICD between the DAACs and ESDIS on what the raw data would contain.

The first data collection began in March 1995 with a few of the DAACs. Gradually, all of them developed the capability to send files in formats that could be used by the SCRS. Many of the DAACs, while not sending data from March 1995, sent back data to build a complete data set so that the SCRS databases are very nearly complete to March 1995. The Fab Four hand statistics collection continued in parallel with the SCRS data until August of 1995, when it was decided to use the SCRS in the future.

## **2.2 Performance Measurement of ECS**

The ECS provides the "core" common capabilities and infrastructure required for performing planning and scheduling, command and control, product generation, information management, data archiving and distribution, and user access to data held by EOSDIS. ECS consists of three segments: the Science Data Processing Segment (SDPS), the Flight Operations Segment, and the Communications and System Management Segment.

The SDPS supports product generation, data archiving and distribution, and information management. The SDPS hardware and software developed as a part of ECS resides and operates at the DAACs. SDPS supports the integration and testing of software for product generation algorithms developed by the EOS investigators. It provides for planning of data product generation, taking into account interdependencies among them, and the distribution of computational resources. It provides for ingest and storage (temporary or permanent, depending on data type) of data sets needed from other data centers to support the generation of standard data products. It generates standard products in a timely manner using the investigator-provided software. It supports the extraction of appropriate subsets of standard data products to assist in scientific quality control by the respective investigators. It supports reprocessing as required.

The SDPS functions may also be provided by a separate system called a Science Investigator-led Processing System (SIPS). In this case, data at one level is sent to the SIPS where it is processed into a higher level and sent back.

The EDGRS is the metrics collection agent in the ECS environment, more specifically the ECS/SDPS at each DAAC. Future enhancements to EDGRS will provide support for other non-ECS sources of metrics. To support ECS metrics, EDGRS must import

information from various data tables in the ECS environment. This information is then used to generate various reports.

The types of information to be reported on by each ECS DAAC include the following general topics:

- Amount of data ingested
- Amount of data archived
- Amount of data retrieved from the archive
- Data distributed to external users

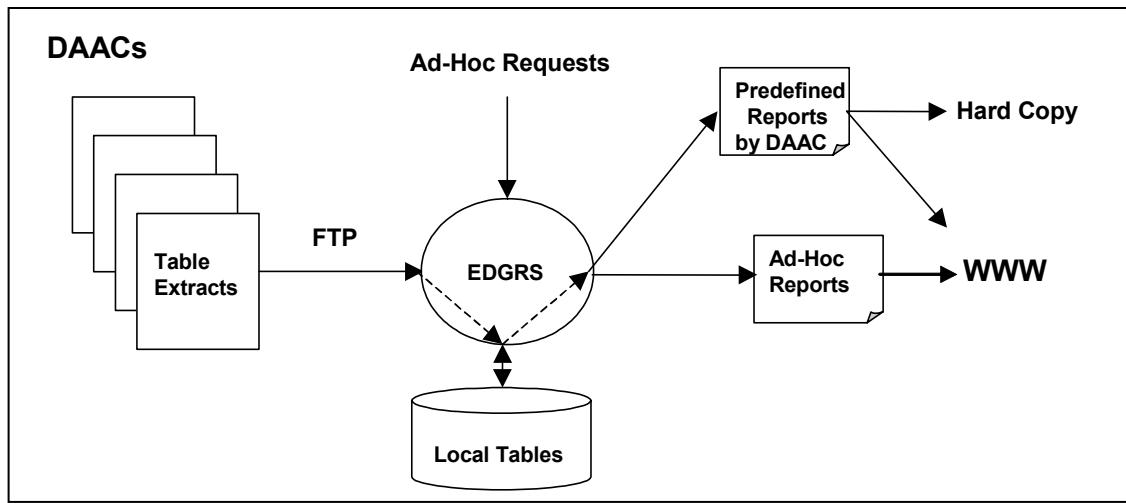
EDGRS also continues to support the non-ECS (V0) metrics collection and includes metrics for this purpose on the following general topics:

- Amount of data archived
- Data distributed to external users

The operations concept for EDGRS is discussed in Section 3. The general topics noted above, and the specific reports to address them, are discussed in Section 4. These topics are those most requested by the SOO to summarize the operations and resource usage at the DAACs.

### 3 Operations Concept for EDGRS

The EDGRS data collection scheme is designed to reside at each DAAC. This collection activity periodically pulls metrics information and FTPs that information to the centralized EDGRS location. Then this data is imported into local tables. Periodically, standardized reports are generated and posted on the WWW. Ad-hoc reports are supported through a Web-based user interface. The reports may be printed locally or produced on the Web. This is demonstrated in the following diagram.



*Figure 3-1 General Data Flow*

The data collection strategy is to acquire more data than is minimally necessary from the metrics data provider to increase the likelihood of being able to support any currently unknown reporting requirements

#### 3.1 Data Collection at DAACs

Data is collected at each DAAC and FTP'd to the central EDGRS location. This approach has the following advantages:

- Data from all of the DAACs are put into a centralized location from which reports can be generated that cross DAAC boundaries.
- Consistent control of the reports and posting of information is done from a single location. This has fewer interfaces and requires less labor to maintain.
- A copy of the data provides an archive of information that is managed separately from the DAAC's installations. This is of value in case reports using the data are requested after the data is deleted, lost, or archived at the DAAC.
- Performance on the DAAC machine is minimized as much as possible since the query to extract the data is only periodic (daily). Ad-hoc querying on the live DAAC machine would have potential resource conflicts.

This approach has the following disadvantages:

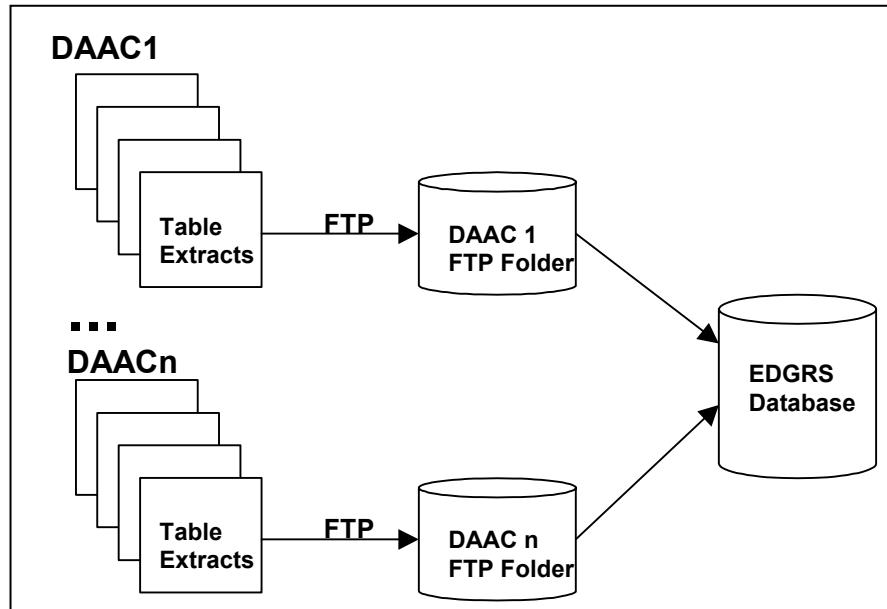
- A copy of the data introduces the risk of not having all of the data needed for making reports or some other similar case of disjoint data between ECS and the central repository.
- There is a time lag between the accessing of data and the posting of reports that is necessitated by the need to first extract the data, transfer the data, archive it, and then report it. The extraction is done periodically (once per day). All of the major operations are individually scheduled events.
- The transfer of data must be monitored daily (Monday - Friday) to ensure that the transfers are proceeding as planned.
- Access to the DAACs tables must be coordinated between personnel at the central location and each ECS system in order to implement the FTP functionality.

To extract data from the ECS tables and send it to the centralized location, the following must occur:

- An account must be set up with the authorization to query (via Sybase SQL embedded into UNIX scripts) the ECS tables and create files in a place from which they can be FTP'd.
- The scripts that pull the data must be installed in this account.
- A UNIX script that initiates the queries must be installed. This script should be set up to initiate automatically on a predefined interval via the 'cron' function or with a script that sets up the next execution automatically.
- A UNIX script that initiates the FTP of the data retrieved in the prior step should be installed to run in a periodic manner as above.

The non-ECS metrics are provided by DAAC-developed software in a form that conforms to the Interface Definition Document (IDD).

### **3.2 Central database**



*Figure 3-2 Database Ingest*

As shown in Figure 3-2, the metrics data is included into files at the DAAC that are then FTP'd to the assigned FTP folder on the system on which EDGRS resides. These files are created as ASCII text files. In most cases, they are set up so one line in the file represents one record in the original ECS table, usually in a fixed width per column fashion.

Once the data is FTP'd to the DAAC FTP folder in the EDGRS environment, it is then imported into the EDGRS database tables. The data is tagged by DAAC, so data queries can look at data for a single DAAC or across DAACs.

This import process is automated so that the FTP and imports run automatically. Daily checks by EDGRS operations personnel can ensure that some data was received. If no data is received from a given DAAC when expected, it is likely that the automated script or FTP capability is down and needs to be restarted. The ECS pull-scripts are designed to pull data with some date overlap so that when some portion of the automation is down, prior days' data can be retrieved. EDGRS removes redundant data.

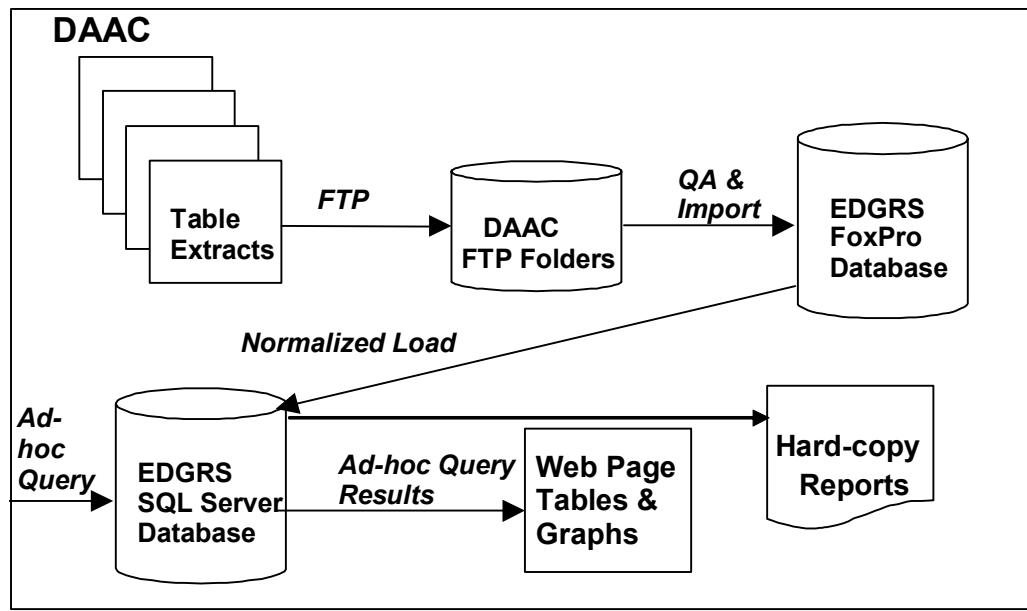
As indicated in Figure 3-2, the same data can reside in various formats in multiple places. This includes the original ECS table, the flat files prior to FTP, the flat files after FTP, and the data in the EDGRS databases. To reduce the duplication and to reserve disk space, the flat text files are archived and deleted after a parameterized time window. Copies of the database will also be made at a parameterized interval to ensure that there is adequate backup of the data.

### **3.3 *The EDGRS Database Environment***

The EDGRS database is implemented in a combined Microsoft FoxPro and Microsoft SQL Server environment. The system was first developed as a prototype (in FoxPro) and finally resides in a SQL Server database structure.

The structure of the FoxPro database is set up to mimic the structure defined by the ECS output scripts. Data fields are QA'd and converted where necessary to provide standard date/time, number, and character width in the resulting EDGRS database tables.

As data began to flow into the databases and estimates to the sizing were refined, the development staff determined that a larger database tool would be desirable. To remain compatible with FoxPro and to maintain the flow of production with the least effort and cost, the staff selected the SQL Server database engine. See Figure 3-3.



**Figure 3-3 Simplified Database Layout**

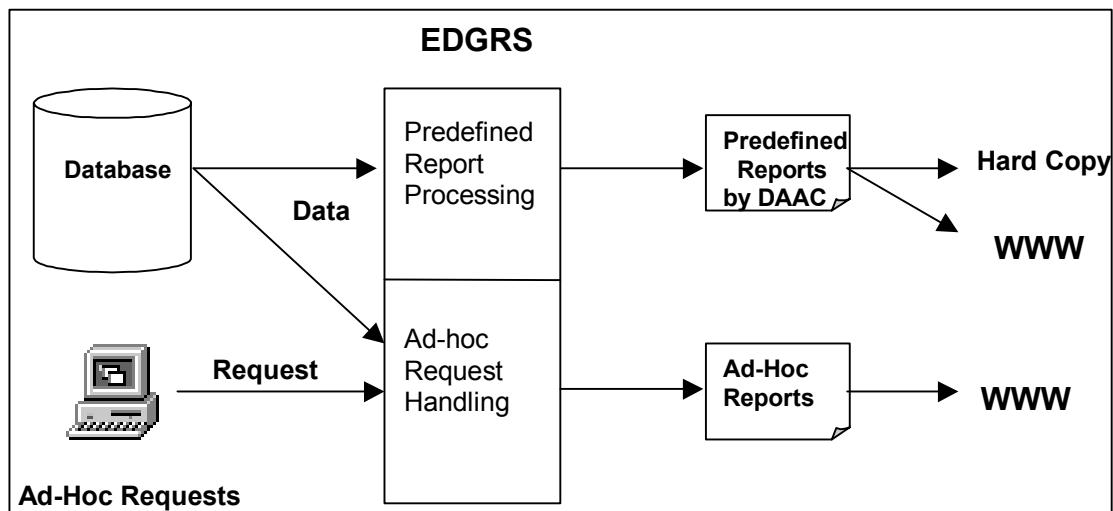
The SQL Server database version of the EDGRS data has provided some normalization to optimize the space used to hold the data. Also, summary tables have been created in SQL Server to speed up the reporting for ad-hoc queries as well as for standard reports and graphs. Graphs are pre-generated and accessible from the web page. Reports other than ad-hoc reports are also pre-generated and accessible from the web page. Hard copy reports are printed for formal reporting such as monthly and annual reports.

### 3.4 Reporting

The general report handling is summarized in Figure 3-4. There are two basic types of reports. These include the predefined and the ad-hoc report types. Predefined reports are reports that have been previously developed, tested, installed, scheduled for periodic execution, cover a defined time range (if applicable), and are in production.

Ad-hoc report requests are reports that are requested by a user, and may control the following parameters:

- The user can specify a time range for the report to cover (predefined report or query).
- The user can select which report to run.



*Figure 3-4 EDGRS Report Processing*

## **4 EDGRS Reports and Graphs**

The types of information to be reported on by each DAAC include the following general topics:

- Amount of data ingested
- Amount of data archived
- Amount of data retrieved from the archive
- Amount and type of data distributed to external users

These reports are organized for the intended audience, as follows:

- EDGRS Annual Report
- EDGRS Web Site Posting - intended for DAAC operations or SOO management, containing detailed metrics and posted on the Web
- Ad-hoc and Special Delivery Reports - intended for any interested party to formulate specific on-line and interactive queries; or, to provide pre-specified reports to individual users.

Data from reports may be presented in one of the following forms:

- Formatted hard-copy outputs. This method is used for pre-defined reports.
- HTML tables with headers. These are viewable from a browser. They can be cut and pasted from the browser into Microsoft Word if using Microsoft Internet Explorer.
- Graphs. Predefined graphs are generated using a java-based tool to access the SQL Server database.

### ***4.1 Annual Report***

The annual report covers the government fiscal year as its reporting period. It provides various tables and graphs to summarize the metrics activity across the DAACs for this fiscal year and over time. It is generated 2-3 months after the end of the fiscal year.

### ***4.2 Ad-hoc and Special Reports***

**Ad-hoc Reports.** Questions arise that the predefined reports cannot answer. For this reason, an ad-hoc query capability via the WWW is provided. This query capability is currently limited to selecting a time range for predefined report layouts. A significant number of reporting options are available using this feature, including selecting data by time range, DAAC, mission, instrument, data type, level, and discipline. The formatting options for the output are HTML formatted tables or tab-text outputs.

The report layouts that are available are described in the EDGRS Ad-hoc Query Results Field Definitions document. This document is available on the EDGRS web site, which is as follows:

**<http://edgrs.gsfc.nasa.gov:8000>**

**Special Reports.** Periodically special studies need to be performed or questions need to be answered based on EDGRS data. The NASA technical contact person will usually request these

studies of the data management team that is managing EDGRS. Each request could result in a one-time report generated from the EDGRS data, data analysis, or reports that need different delivery methods (i.e., not WWW postings).

As the data management team receives these requests, they will be coordinated with the NASA technical contact person and scheduled for implementation.

## **5 Database Design**

After a discussion of database segmentation and summary tables, this section describes the database design and special processing for each subsystem handled by EDGRS. Each subsystem contains the following components:

- Subsystem overview – describes the general approach to organizing and processing the data.
- Table organization – describes the tables and views and indexes used and any normalization approaches employed.
- Data flow diagrams – describe the flow of data through the various tables, including references to the file names and procedures used.
- Other notes or comments pertinent to the subsystem being described.

The detailed field definitions are found in ‘Release 1 EOSDIS Data Gathering and Reporting System (EDGRS) Database Design and Database Schema Specifications’.

### **5.1 *Database Segmentation***

#### **5.1.1 Description**

The larger base tables in the EDGRS SQL Server database, i.e. those with more than several million records, are split into special segments which can be recombined into views that function like the original table in database updates, insertions and deletions. For these ‘partitioned’ views to work this way, the segments and views must adhere to the following rules:

- The segments must have the same structure including the primary key
- They must contain a field, the partition field, that can be used to define the range of records contained in each segment
- There must be a check constraint on the partition field with specific values or ranges that do not overlap
- The partition field must be part of the primary key
- The ‘partitioned’ view must include all segment fields or columns
- Columns and segments cannot be duplicated in the ‘partitioned’ views

In addition, the following requirement applies to the current or latest segments in the EdgrsCurrent database

- At least 45 days of data should remain in the latest segment after the split. This permits the use of the ‘last30days’ method for generating summary tables from base tables (See Section 5.2) and takes care of the n-day window used for adding new records (where n is less than 45). It also allows for the difference between the time fields used for segmentation from those used for generating the summary tables (See Section 5.2.5).

#### **5.1.2 Why Use Segmentation?**

Using database segmentation has the following advantages:

- Smaller tables for segments facilitate backfill and other maintenance functions applied to the base table (including backups and providing flexibility in allocation of tables to disk drives)
- The segments composing the views can reside in different databases or disks. This helps to limit the size of each database and thus facilitates their backup.
- Only the affected segments needed for a query will be used if the partition field is included in the where clause.

### 5.1.3 Limitations

The ‘partitioned’ views have certain limitations in usage. The major one is that they cannot appear more than once in the same query. For example, they cannot be used in self-joins. The limitation can be overcome by using intermediate tables in performing the queries. Another limitation is that the segmentation procedures must be performed periodically. Currently those procedures are performed every 6 months. Those local operational procedures are described in items #5 and #7 in the Edgrs HowDoI document on samana in the Operations/EDGRS Procedures directory.

### 5.1.4 Offline vs Online Segments

In order to allow older segments to be moved offline so that more disk space is available, we distinguish between segments by means of views. Older segments that can be moved offline are placed in ‘old’ views, i.e. the view names end with ‘\_old’, and segments that must remain online are placed in ‘new’ views, i.e. the view names end with ‘\_new’. The segments in these views are treated differently when they are used to generate summary or other derived tables. The part of a derived table associated with offline segments is generated only once either after some of the segments of the corresponding base table has been designated as offline or after a backfill (to correct older data) of the corresponding base table that affects the offline segments has been done (See HowDoI.doc items #9 and 14). On the other hand, the part of the derived table associated with online segments is regenerated daily through scheduled EDGRS production processing except for distribution and ingest summary tables when the last30day method is used (See Section 5.2.2 below). These derived tables and the views they depend on are shown in Table 5-1 below.

**Table 5-1 Tables Affected by Distinguishing Between Online and Offline Segments**

Base Table	View Pair (old/new)	Derived Table
inreqdata	view_inreqdata_old/new	rp_ingest_instrument_datatype
SCRS_ftp	view_scrs_ftp_old/new	scrs_rp_ftp_summary
SCRS_ftp	view_scrs_ftp_old/new	scrs_rp_ftp_summary_by_daac
SCRS_wdd	view_scrs_wdd_old/new	scrs_rp_wdd_summary
SCRS_wdd	view_scrs_wdd_old/new	scrs_rp_wdd_summary_by_daac
SCRS_www	view_scrs_www_old/new	scrs_rp_www_summary
SCRS_www	view_scrs_www_old/new	scrs_rp_www_summary_by_daac
dist	view_dist_old/new	rp_dfa_instrument_datatype
dist	view_dist_old/new	rp_dfa_by_subscription
dist	view_dist_old/new	rp_dtu_summary
dist	view_dist_old/new	rp_dtu_instrument

<b>Base Table</b>	<b>View Pair (old/new)</b>	<b>Derived Table</b>
dist	view_dist_old/new	rp_order_status
dist	view_dist_old/new	rp_dtu_orders
dist	view_dist_old/new	orderspancnt
dist	view_dist_old/new	orderspandetail

TBS - Add slides 20-21 from EDGRSOOverview2004.ppt.after modifications.

### 5.1.5 Subsystems and Tables Affected

Different modes of segmentation are used for different base tables depending on the main database function needs for those tables and the extent to which these needs can benefit from the segmentation process. Table 5-2 shows the base tables that have been segmented and the segmentation method.

**Table 5-2 Base Table Segmentation in EDGRS**

<b>Subsystem</b>	<b>Base Table</b>	<b>Partition Field</b>	<b>DB Functions</b>	<b>Same DB</b>	<b>Seg Method</b>
Archive	dsmgdran	inserttime	Full range update Full range insert Query	Y (See Note 1)	HowDol - #5
Ingest	inreqdata	procstart	Current seg update Current seg insert Query	N	HowDol - #7
Dsdd	dsddgran	starttime	Current seg update Current seg insert Query	N	HowDol - #7
	dsddrqst	starttime	Online seg update Curr seg insert Query	N	HowDol - #7
Distrib	dist	starttime	Backfill Current seg update Current seg insert Query	Y (See Note 2)	HowDol - #7
Scrs	SCRS_ftp	ftp_dt	Current seg update Current seg insert Query	N	HowDol - #7
	SCRS_wdd	www_dt	Current seg update Current seg insert Query	N	HowDol - #7
	SCRS_www	www_dt	Current seg update Current seg insert Query	N	HowDol - #7

Notes:

1. All segments are now created in the same database, namely EdgrsArchive. New segments will be placed in a different database, e.g. EdgrsArchive01, when the current database becomes too large.
2. All new segments split off from the current segment are now put into the EdgrsDist database. New segments will be placed in a different database, e.g. EdgrsDist01, when the current database becomes too large.

## 5.2 Summary Tables

### 5.2.1 Description and Justification

Summary tables are generated from base tables containing EDGRS data in order to speed up the generation of ad-hoc reports and graphs based on these base tables. Two main types of summary tables exist. One is an upper-level summary table always grouped by metrics provider (e.g. the DAAC) and sometimes by another characteristic of the data such as instrument; and the other is a detail level summary grouped by provider, date and other characteristics of the data. The detail level summary tables are daily tables because each record contains a particular statistic summed over a particular day. The two types of segments associated with a segmented database table are processed separately to generate an ‘old’ part and a ‘new’ part for each summary table. Each part is handled differently as described in Section 5.1.4. Both parts are stored in intermediate tables. The ‘old’ part is generated manually when first created or whenever the old segments of the segmented database table are modified. The ‘new’ part is regenerated or modified during daily processing. The two parts are then joined together via views to provide access to the entire summary table when deriving other tables or generating reports. This use of intermediate tables ensures minimum impact on the existing base summary tables during scheduled daily processing.

### 5.2.2 Last 30 Days Method

In this method of generating summary tables, only the latest 30 days of the summary table are modified thus speeding up the generation process. This is done by first deleting the last 30 days of data from the summary table, regenerating the last 30 days from the current segment in the base table and inserting those records into the summary table. Currently, this is only done for 3 summary tables and only on weekdays (to speed up processing). On the weekends, full summary table generation is performed, allowing changes to parts of the current segments older than 30 days, e.g. from backfill, to be picked up then. The summary tables affected are shown in Table 5-3 below.

**Table 5-3 Summary Tables Using the Last30days Method**

Subsystem	Base Table	Summary Tables	Seg Time Used	Summary Date1
Ingest	inreqdata	rp_ingest_instrument_datatype	procstart	sortdate
Distribution	dist	rp_dfa_instrument_datatype rp_dfa_by_subscription	starttime	sortendtime

1. Summary Date is the date field used in generating the corresponding summary table:  
sortdate is based on procend and sortendtime is based on endtime.

### 5.2.3 Correction for Null ordered Values

A complication arises when a summary table contains counts of orderid and the base table contains orderid values that are null. Preventing the counting of null orderid values in this case requires that a separate pass be made in getting the number of orders other than the one used to get the request statistics. The results of the two passes are stored in separate intermediate tables designated ‘ord’ and ‘req’ for orders and requests, respectively, and the two tables are merged

appropriately to generate the corresponding ‘old’ or ‘new’ parts of the summary table. The summary tables affected by this correction are shown in Table 5-4 below.

**Table 5-4 Summary Tables Affected by Null Orderids**

Subsystem	Base Table	Summary table
Distribution	dist	rp_dfa_instrument_datatype
Distribution	dist	rp_dtu_instrument
Distribution	dist	rp_dtu_summary

#### 5.2.4 Correction for non-null orderids spanning multiple days

Another problem arises when the summary tables generated from the dist base table are used to generate ad-hoc reports that display the number of orders. Because orders can span multiple days, any summation over multiple days based on a daily summary table can result in overcounting of orders when the report is generated. Correcting this error is done by using the orderspandetail table, which contains the detail records from the dist base table that cause the problem, to calculate the error and subtract it from the erroneous result when the report is generated (See Appendix D). The orderspandetail table itself is generated in two parts. One part comes from the offline segments and is generated at the time the old portions of the dist summary tables are built. The other comes from the online segment and is generated during daily processing. The summary reports affected and the summary tables from which they are generated are shown in Table 5-5 below.

**Table 5-5 Summary Reports Affected by Orders Spanning Multiple Days**

Base Table	Summary Table	Summary Report
dist	rp_dtu_instrument	DtuInstrument
dist	rp_dtu_instrument	DtuDatatype
dist	rp_dtu_summary	DtuSummary

#### 5.2.5 Difference between time fields used for segmentation and time fields used for summary reports

The field chosen to perform the segmentations on the EDGRS base tables was the time field used in performing the daily Data Transformation Services (DTS) tool to transfer from FoxPro to SQL Server. This is generally the start time associated with that type of data. On the other hand, reports generally use end times, which occur later than start times, to display the data. This difference can cause report errors if individual segments rather than base table views are used to generate those reports. The different time fields used for segmentation and summary report generation are shown in Table 5-6 below.

**Table 5-6 Different Time Fields Used for Segmentation and Report Generation**

Base Table	Segmentation Time Field	Summary Report Time Field
dsmdgran	inserttime	sortdate – based on inserttime
dsmdgran	inserttime	sortobsdate – based on begintime

<b>Base Table</b>	<b>Segmentation Time Field</b>	<b>Summary Report Time Field</b>
inreqdata	procstart	sortdate - based on procend
dist	starttime	sortendtime - based on endtime
SCRS_ftp	ftp_dt	ftp_dt
SCRS_wdd	www_dt	www_dt
SCRS.www	www_dt	www_dt

To avoid the above problem, stored procedures or queries should use views of the base table that overlap the time range specified in a report by a reasonable time, e.g. 30 days or more, or use the entire base table. The problem should not occur when summary tables are used to generate reports.

### **5.3 EDGRS Ingest Subsystem**

The EDGRS ingest subsystem captures/reports metrics on the data granules or other products received at the data centers from ECS sources. The ingest data received is in two forms. One form is associated with individual ingest requests and is stored in the inreqhdr table. The other form is associated with the actual data received and is stored in the inreqdata table. Both forms of data are received by the FoxPro database but selected data from inreqhdr and inreqdata tables are merged when transferred to the inreqdata table in the SQL Server database.

The inreqdata table is segmented into multiple tables. This is done for both performance and storage allocation reasons. There are views named ‘view\_inreqdata’ and ‘view\_inreqdata\_new’ which link the table or parts of it together for queries. The segmentation procedure is described in Item #7 of the ‘HowDoI.doc’ file and must be done periodically, currently every six months.

The data flow diagrams, procedure/view definitions, table definitions and index definitions are described below.

Notes:

- All procedures and tables described below are in the EdgrsCurrent database except where indicated otherwise.
- The field definitions for the tables described here appear separately in the document ‘Release 1 EOSDIS Data Gathering and Reporting System (EDGRS) Database Design and Database Schema Specifications’.
- No triggers are used in this subsystem.

## FoxPro Database

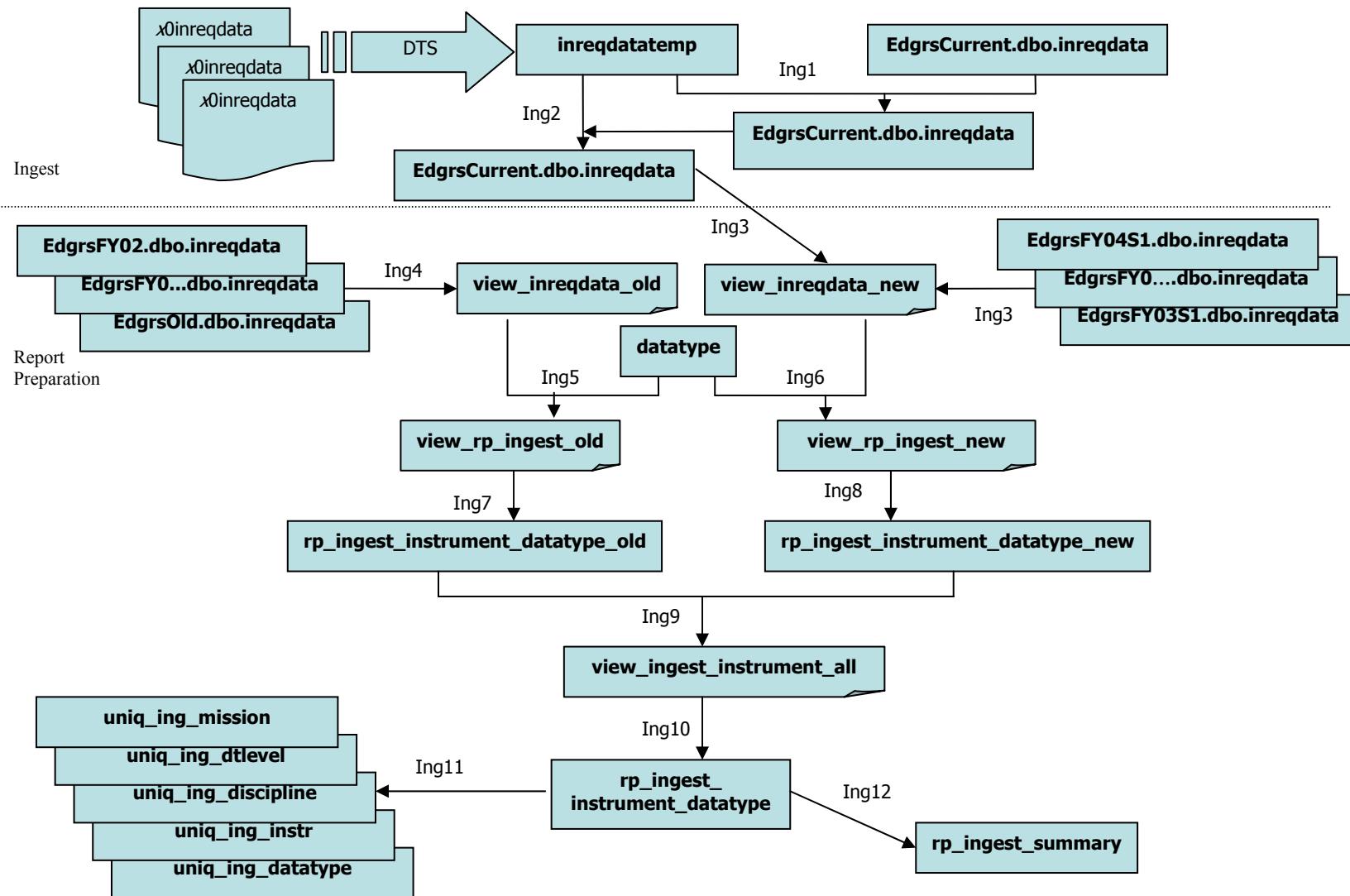


Figure 5-1 Ingest Data Flow Diagram (DFD)

**Table 5-7 Ingest DFD Key**

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
Ing1	pro_update_inreqdata	Update/Update_inreqdata.sql	Update existing records in the current segment	Derive sortdate from procend.
Ing2	pro_update_inreqdata	Update/Update_inreqdata.sql	Copy new records to the current segment	Derive sortdate from procend
Ing3	view_inreqdata_new	Paul/ (should be CreateTableStatements/Segmentation/ ) Create_inreqdata_views.sql	Group online segments via UNION ALL	View allows tables to be split for performance.
Ing4	view_inreqdata_old	Paul/ (should be CreateTableStatements/Segmentation/ ) Create_inreqdata_views.sql	Group offline segments via UNION ALL	View allows tables to be split for performance.
Ing5	view_rp_ingest_old	Report/Sps0_ingest_prod_curr_old.sql	Accumulate count of ingest requests and data granules for new offline data	Includes only data with state = 'ARCHIVED' or 'SUCCESSFUL'. Uses sortdate.
Ing6	view_rp_ingest_new	Report/Sps0_ingest_prod_curr_new.sql	Accumulate count of ingest requests and data granules for new online data	Includes only data with state = 'ARCHIVED' or 'SUCCESSFUL'. Uses sortdate.
Ing7	pro_rp_ingest_instrument_old	Report/Sps0_ingest_prod_curr_old.sql	Regenerate old part of summary table.  Only run once.	Uses datatype table to extract mission, level, instrument, discipline info. Values set to 'UNKNOWN' if NULL
Ing8	pro_rp_ingest_instrument_new	Report/Sps0_ingest_prod_curr_new.sql	Regenerate new part of summary table. Run daily on weekends for full summary table build.	Uses datatype table to extract mission, level, instrument, discipline info. Values set to 'UNKNOWN' if NULL
Ing9	view_ingest_instrument_all	Report/Sps0_ingest_prod_curr_all.sql	Accesses old and new parts of summary table via UNION ALL	.

<b>ID</b>	<b>Procedure/View</b>	<b>File Defining Procedure/View</b>	<b>Action / Key for Relation</b>	<b>Comments</b>
Ing10	pro_rp_ingest_instrument_all	Report/Sps0_ingest_prod_curr_all.sql	Generate rp_ingest_instrument_datatype table from view	
Ing11	pro_rp_ingest_instrument_all	Report/Sps0_ingest_prod_curr_all.sql	Generate uniq_ ing tables from rp_ingest_instrument_datatype	.
Ing12	pro_rp_ingest_summary_all	Report/Sps0_ingest_prod_curr_all.sql	Generate rp_ingest_summary table from rp_ingest_instrument_datatype	

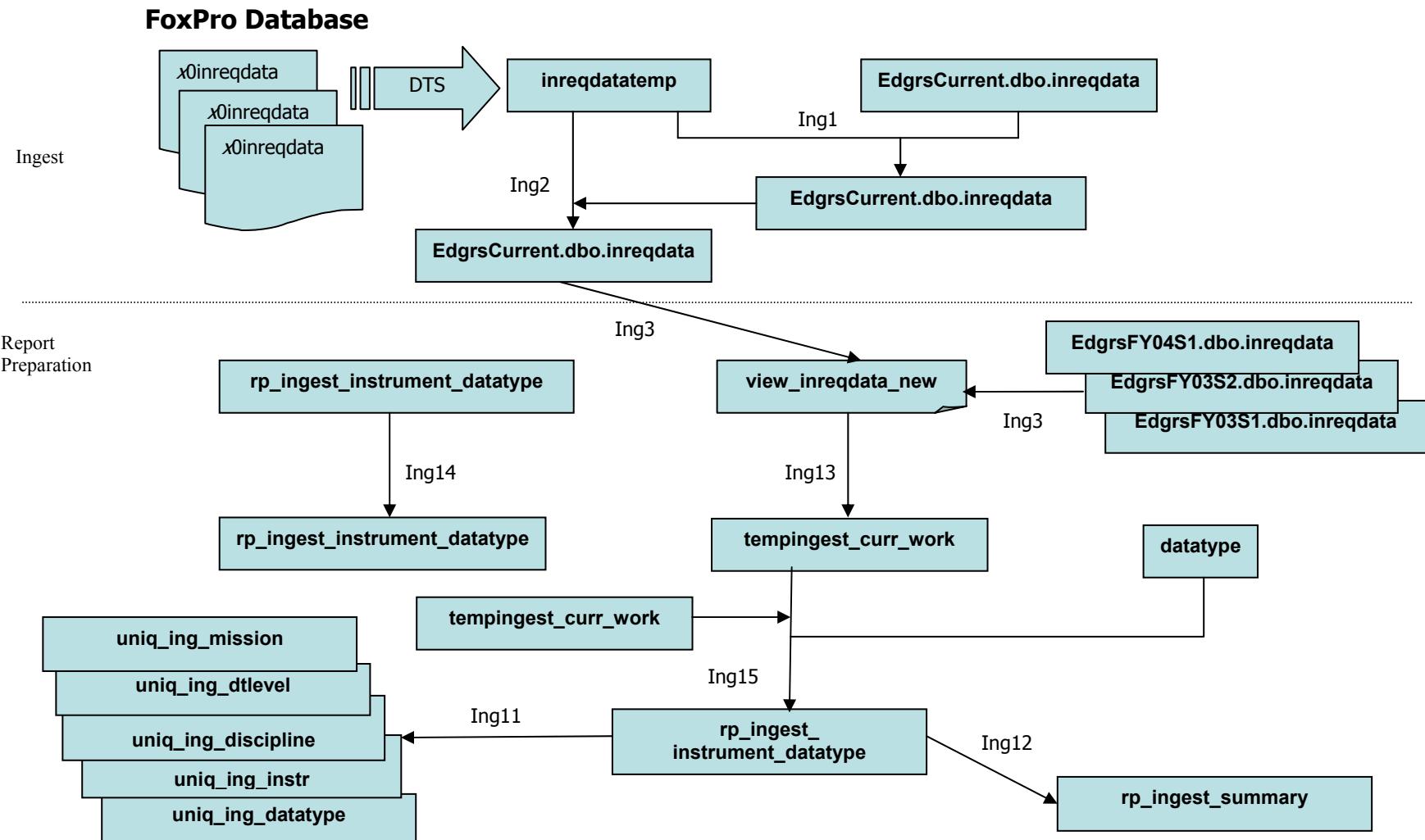


Figure 5-2 Ingest – Last 30 Days DFD

**Table 5-8 Last 30 Days DFD Key**

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
Ing1	pro_update_inreqdata	Update/Update_inreqdata.sql	Update existing records in the current segment	Derive sortdate from procend.
Ing2	pro_update_inreqdata	Update/Update_inreqdata.sql	Copy new records to the current segment	Derive sortdate from procend
Ing3	view_inreqdata_new	Paul/ (should be CreateTable Statements/Segmentation/) Create_inreqdata_views.sql	Group online segments via UNION ALL	view_inreqdata_new is used instead of just the current segment to minimize errors that could arise at the segment boundary because the boundary depends on procstart rather than procend.
Ing13	pro_rp_ingest_instrument_last30days_all	Paul/ (should be Report/) Sps0_ingest_prod_last30days_all.sql	Extract the last 30 days of data from inreqdata online segments using sortdate and store it in a temporary table	Includes only data with state = 'ARCHIVED' or 'SUCCESSFUL' and sortdate within last 30 days
Ing14	pro_rp_ingest_instrument_last30days_all	Paul/ (should be Report/) Sps0_ingest_prod_last30days_all.sql	Delete the last 30 days of data from the current content of table rp_ingest_instrument_datatype using sortdate.	
Ing15	pro_rp_ingest_instrument_last30days_all	Paul/ (should be Report/) Sps0_ingest_prod_last30days_all.sql	Generate the last 30 days portion of the summary table rp_ingest_instrument_datatype from view_inreqdata using sortdate and add the rebuilt part back to the summary table.	Uses datatype table to extract mission, level, instrument, discipline info. Values set to 'UNKNOWN' if NULL
Ing11	pro_rp_ingest_instrument_all	Paul/ (should be Report/) Sps0_ingest_prod_curr_all.sql	Generate uniq_ingroup tables	.

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
Ing12	pro_rp_ingest_summary_all	Paul/ (should be Report/) Sps0_ingest_prod_curr_all.sql	Generate rp_ingest_summary table	

**Table 5-9 Ingest Subsystem Table Definitions**

Table ID	EDGRS Table Name	Primary Key	EDGRS Table Description	Comments
1	inreqdatatemp	None	staging area for ingesting metrics into the system for ingest data. .	The structure of this file mimics the structure of the input data. Data is moved into this area and then additional processing is performed before moving the data into the inreqdata table.
2	inreqdata	provider, requestid, granuleid, procstart	This table contains the segments of the inreqdata table and contains one record for each granule/product stored in the database.	Each segment resides in a different database. The current segment is in the EdgrsCurrent database. Other segments lie in the EdgrsFY04S1, EdgrsFY03S2, EdgrsFY03S1 databases  All of these segments have the same structure but different check constraints on procstart.
3	datatype	datatypeid  Or  esdt_id	The datatype table is used to uniquely map the ESDT names to mission, instrument, level, and discipline for reporting purposes.	Datatypeid is the ESDT name used by the DAAC. Esdt_id is an internally coded value used by EDGRS. Either field may be used to uniquely identify or link a datatype. Note that the only reliably completed fields in this table are esdt_id, datatypeid, descriptio (description text), edginstrum (the instrument), mission, level, and discipline. Other fields were used in initial data gathering efforts and have no guarantee of useful information.  Note. This table is maintained manually.
4	rp_ingest_instrument — datatype_old rp_ingest_instrument — datatype_new	None	Old and new parts of the rp_ingest_instrument_datatype table	Both tables have the same fields as the full summary table.
5	rp_ingest_instrument — datatype	None	The rp_ingest_instrument_datatype table is the prime summary table from which ingest reporting is done on the ad-hoc screen, with	

Table ID	EDGRS Table Name	Primary Key	EDGRS Table Description	Comments
			records defined for each date. There is one record for each combination of provider and datatype for each date. The date is based on sortdate which is the procend for ingest of the granule.	
6	rp_ingest_summary	None	The rp_ingest_summary table reports counts of granules and volume for each data provider . The insertime is based on the ingested granule's procend time.	Built from rp_ingest_instrument_datatype table. Currently only ECS DAACs. Supports display in column format.
7	uniq_ing_mission	None	Table of distinct missions used in the data.	Could be used in driving pull-downs on web pages
8	uniq_ing_dplevel	None	Table of distinct levels used in the data	Could be used in driving pull-downs on web pages
9	uniq_ing_discipline	None	Table of distinct disciplines used in the data	Could be used in driving pull-downs on web pages
10	uniq_ing_instr	None	Table of distinct instruments used in the data	Could be used in driving pull-downs on web pages
11	uniq_ing_datatype	None	Table of distinct datatype names used in the data	Could be used in driving pull-downs on web pages

**Table 5-10 Ingest Subsystem Index Definitions**

Index ID	EDGRS Index Name	Table	EDGRS Index Description	Clust-ered	Unique Index	Comments
1	PK_inreqdata	<DB-name>.dbo.inreqdata	Primary Key to inreqdata tables, one for each segment.	Yes	Yes	Provider, requestid, granuleid, procstart
2	INDX_sortdate	<DB-name>.dbo.inreqdata	Sorts data by procend date	No	No	sortdate
3	PK_datatype	datatype		Yes	Yes	datatypeid
4	INDX_insertime	rp_ingest_instrument_datatype	indexed on insertime for speed	Yes	No	insertime
5	index_insertime	rp_ingest_summary	indexed on insertime for speed	Yes	No	insertime

## **5.4 EDGRS Archive Subsystem**

The EDGRS archive subsystem captures/reports metrics on the data granules or other products stored at the data centers that are available for distribution or other uses.

The granules table is segmented into multiple tables. This is done for both performance and storage allocation reasons. There is a view named ‘view\_dsmgdran’ which links the table together for queries.

The data flow diagrams, procedure/view definitions, table definitions and index definitions for the archive subsystem are described below.

Notes:

- No triggers were used in this subsystem.

## FoxPro Database

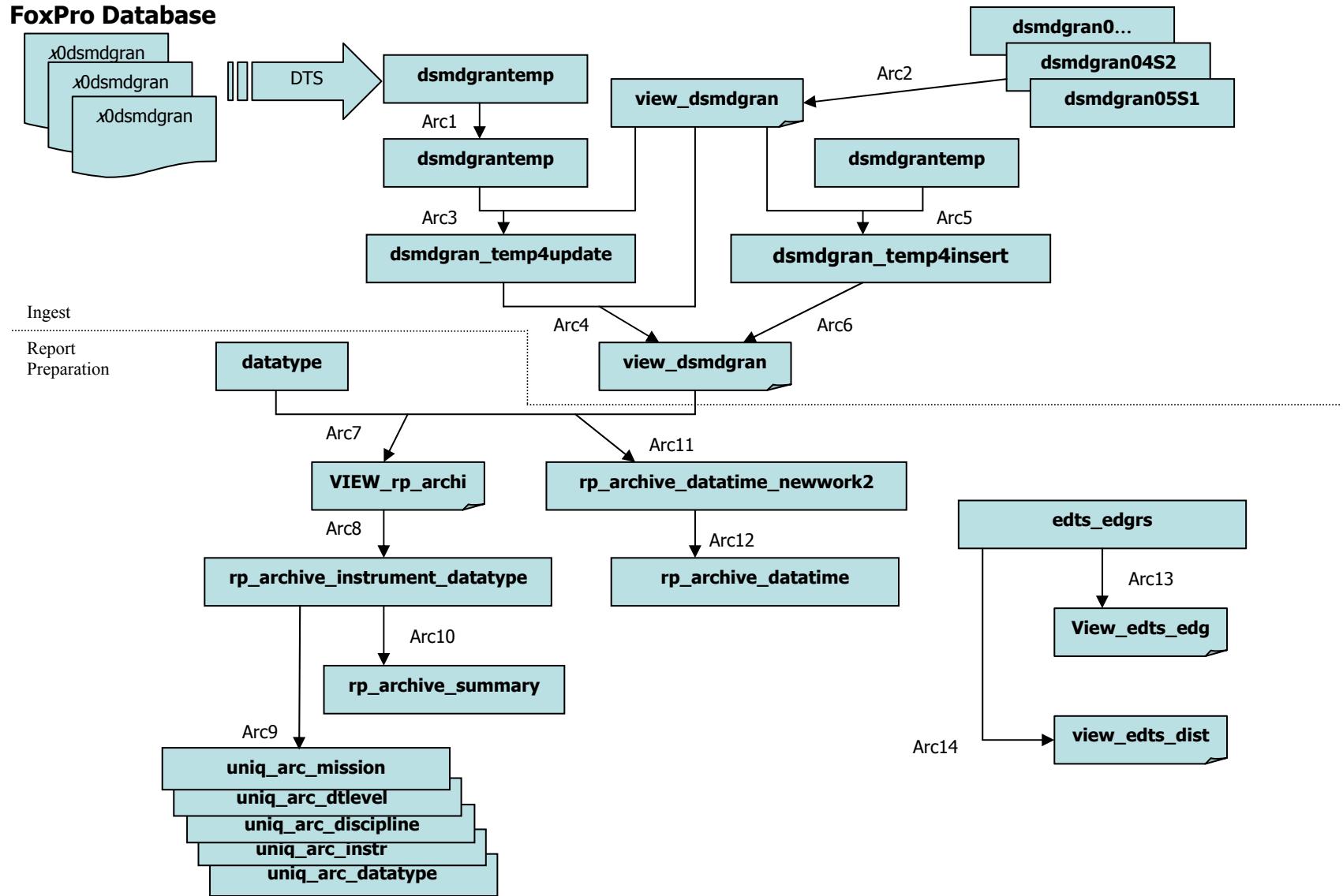


Figure 5-3 Archive – Full DFD

**Table 5-11 Full DFD Key**

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
Arc1	pro_update_dsmgdran	Update_dsmgdran_archive.sql	Update the sizembecs, shortname and delfromArc fields	In dsmdgrantemp Force L70RWRS size to 0; Update sizembecs Upcase shortname, set delfromArc flag
Arc2	view_dsmgdran	Create_dsmgdran_view.sql	Is represented in multiple tables (all fields)	View allows tables to be segmented for performance.
Arc3	pro_update_dsmgdran	Update_dsmgdran_archive.sql	Copy records already in view_dsmgdran / provider+dbid+shortname into table dsmgdran_temp4update	Only copy records that have changes
Arc4	pro_update_dsmgdran	Update_dsmgdran_archive.sql	Update records already in dsmgdran / provider+dbid+shortname with dsmgdran_temp4update	Only update records that have changes
Arc5	pro_update_dsmgdran	Update_dsmgdran_archive.sql	Copy records not yet in view_dsmgdran / provider+dbid+shortname into table dsmgdran_temp4insert	
Arc6	pro_update_dsmgdran	Update_dsmgdran_archive.sql	Insert the new records into view_dsmgdran / provider+dbid+shortname from table dsmgdran_temp4insert	
Arc7	VIEW_rp_archive	Sps0_archive_prod.sql	Create view where /datatype.datatypeid= view_dsmgdran.shortname	Defines other/unknown values for instrument, mission, level, discipline where no map found in datatype table. View sums granule counts and volume.
Arc8	pro_rp_archive_instrument	Sps0_archive_prod.sql	Move all records into a temp table and then finally into the summary table rp_archive_instrument_datatype	Uses temporary table #rp_archive_instrument_datatype_newwork. Temp table is used to not slow usage of summary table.
Arc9	pro_rp_archive_instrument	Sps0_archive_prod.sql	Create tables of distinct (unique) mission, level, discipline, instrument, and datatype	
Arc10	pro_rp_archive_summary	Sps0_archive_prod.sql	Create table rp_archive_summary based on /provider+instrument+deleted+insertime	Puts outputs in columns for display purposes.

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
Arc11	pro_rp_archive_datatime	Sps0_archive_datatime_prod_new	Create temp table rp_archive_datatime_newwork2 where /datatype.datatypeid= view_dsmgdn.shortname	Uses sortobsdate (data time) instead of processing time.
Arc12	pro_rp_archive_datatime	Sps0_archive_datatime_prod_new	Move all records into a temp table and then finally into the summary table	Uses temporary table rp_archive_datatime_newwork2. Temp table is used to not slow usage of summary table.
Arc13	view_edts_edgrs	edts.sql	Create view where /dsmgdn.datatypeid= edts_edgrs.shortname	Extracts from view_dsmgdn the datatypes defined in edts_edgrs

**Table 5-12 Archive Subsystem Table Definitions**

Table ID	EDGRS Table Name	Primary Key	EDGRS Table Description	Comments
1	dsmgdnrtemp	Provider, dbid, shortname	The dsmgdnrtemp table is a staging area for ingesting metrics into the system for archive data.	The structure of this file mimics the structure of the input data. Data is moved into this area and then additional processing is performed before moving the data into the dsmgdn table.
2	dsmgdn_temp4update	Provider, dbid, shortname	The dsmgdn_temp4insert and dsmgdn_temp4update tables are temporary tables. The structure of the latter is identical to dsmgdnrtemp. The structure of the former is similar to that of the dsmgdn segments.	This processing involves the use of the dsmgdn_temp4insert and dsmgdn_temp4update tables as temporary holding areas.
3	dsmgdn_temp4insert	None		.
4	dsmgdn dsmgdn_old dsmgdn_00 dsmgdn_01 dsmgdn_02S1 dsmgdn_02S2 dsmgdn_03S1 dsmgdn_03S2 dsmgdn_04S1 dsmgdn_04S2	Provider, dbid, shortname, inserttime	These tables are the segments of the dsmgdn table which contain one record for each granule/product stored in the database.	Inserttime is not required to access records uniquely. It is used for table segmentation. dsmgdn is an empty table showing the structure and is not included in the view.  All of these tables have the same structure.
5	datatype	datatypeid	The datatype table is used to uniquely map the	Datatypeid is the ESDT name used by the DAAC. Esdt_id is

Table ID	EDGRS Table Name	Primary Key	EDGRS Table Description	Comments
		Or esdt_id	ESDT names to mission, instrument, level, and discipline for reporting purposes.	<p>an internal coded values used by EDGRS. Either field may be used to uniquely identify or link a datatype. Note that the only reliably completed fields in this table are esdt_id, datatypeid, descriptio (description text), edginstrum (the instrument), mission, level, and discipline. Other fields were used in initial data gathering efforts and have no guarantee of useful information.</p> <p>Note. This table is maintained manually.</p>
6	rp_archive_instrument_datatype		The rp_archive_instrument_datatype table is the prime summary table from which archive reporting is done on the ad-hoc screen. There is one record for each combination of provider, datatype, deleteflag, and version for each date. The date is based on sortdate which is the insert time for the granule.	
7	rp_archive_summary		The rp_archive_summary table reports counts of granules and volume for each data provider . The insertime is based on the granule insert time in the archive.	Built from rp_archive_instrument_datatype table. Currently only ECS DAACs plus Latis. Supports display in column format.
8	rp_archive_datatime_newwork2		This is a temporary table used in the construction of the rp_archive_datatime.	This table uses the data based on sortobsdate which is the data begin-time.
9	rp_archive_datatime		The rp_archive_datatime table is used for observation time based reporting.	The observation time is based on the begintime.
10	uniq_arc_mission		Table of distinct missions used in the data.	Could be used in driving pull-downs on web pages
11	uniq_arc_dtlevel		Table of distinct levels used in the data	Could be used in driving pull-downs on web pages
12	uniq_arc_discipline		Table of distinct disciplines used in the data	Could be used in driving pull-downs on web pages
13	uniq_arc_instr		Table of distinct instruments used in the data	Could be used in driving pull-downs on web pages
14	uniq_arc_datatype		Table of distinct datatype names used in the data	Could be used in driving pull-downs on web pages
15	edts_edgrs		Contains a list of ESDTs to include in EDTS processing.	Supports view_edts_edgrs used by EDTS organization

**Table 5-13 Archive Subsystem Index Definitions**

Index ID	EDGRS Index Name	Table	EDGRS Index Description	Clust-ered	Unique Index	Comments
1	PK_dsmgdn<segment-name>	dsmgdn_<segment-name>	Primary Key to DsMdGran tables, one for each segment.	Yes	Yes	Provider, dbid, shortname
2	INDX_sortdate_dsmgdn<segm ent-name>	dsmgdn_<segment-name>	Sorts data by processing date	No	No	Processing date=sortdate
3	INDX_sortobsdate_dsmgdn<s egment-name>	dsmgdn_<segment-name>	Sorts data by observation date	No	No	Observation date=sortobsdate
4	archive_<segment-name>dbid	dsmgdn_<segment-name>	Sorts data by dbid	No	No	dbid
5	PK_dsmgdntemp	dsmgdntemp	Primary Key	Yes	Yes	Provider, dbid, shortname
6	PK_dsmgdn_temp4update	dsmgdn_temp4update	Primary Key	Yes	Yes	Provider, dbid, shortname
7	PK_datatype	datatype	Primary Key	No	Yes	datatypeid

## 5.5 EDGRS Distribution Subsystem

The EDGRS distribution subsystem captures/reports metrics on the data granules or other products distributed from the archive (for ECS systems) and distributed to end-users.

Certain distribution tables are segmented into multiple tables. This is done for both performance and storage allocation reasons. There are views defined which link the tables together.

The data flow diagrams, procedure/view definitions, table definitions, index definitions, and field definitions are described below.

No triggers were used in this subsystem.

### 5.5.1 Overview

The distribution subsystem collects metrics on various methods of data distribution and data access from two different types of systems ECS and non-ECS systems. A metrics set is collected from each system type; so, if a given data provider (DAAC) has both ECS and non-ECS systems, then there are two sets of metrics collected and identified for that provider. They are distinguished in the data by different provider names.

The various methods of data distribution are as follows:

- Orders / Subscriptions
- Data Pools (ECS Systems)
- Anonymous FTP (nonECS systems)

- Web Downloads (WDD) (nonECS systems)
- Off-line Inquiries (INQ) (nonECS systems)

Accesses are determined by email address and last name (when available).

Metrics are collected for each type of access/distribution method at the transaction level. A transaction could be an instance of an order, subscription, ftp push/pull, web download, etc. Since there are millions of transactions, reporting on this volume in an ad-hoc fashion would be sluggish at best. To provide a quick access to summarized data, summary tables are pre-built and then made accessible to the ad-hoc capability. Other performance and space optimizations are made by normalizing some of the data upon input. The tables holding the detailed transaction records are referred to as ‘base tables’. Tables holding summarized information are referred to as ‘summary tables’. Summary table names begin with the letters ‘rp\_’. Multiple summary tables are built to support specific reports or report groups.

Figure 5-4 on the following pages demonstrates an overview of the base and summary tables and indicate the data flows to support those tables. The ingesting, normalization, and summary table build processes require many temporary tables. These temporary tables are not described in these figures, but they are identified in later figures and tables in this document. Table 5-14 identifies the names of the base tables supporting the various distribution/access methods.

**Table 5-14 Mapping of Distribution Methods to Tables used in the Data Import Process**

Distribution Method	EDGRS Base Tables	Comments
Distribute-from-Archive (non-End-User) (ECS ONLY)	dsddrqst, dsddgran, dist, subscrpt	Non-End-User distributions are used for production processing, QA, Test, and other local purposes. These purposes are separated using the usertype field in the Dist table and are calculated based on userids and the subscrpt table.
Orders	dsddrqst, dsddgran, products, dist, subscrpt, ecacorder, ecacrqst, users, OMRequest, OMRequestGranule, OMGranule, OMFile, affil, medias	Orders are identified as records that have an order identifier AND/OR a usertype flag in the Dist table that indicates this is an end-user order. Userids in the subscrpt table can be defined as NOT end-user.
Subscriptions	dsddrqst, dsddgran, dist, OMRequest, OMRequestGranule,	A subscription is defined as a routine transfer of data based on a set of criteria that automatically transfers the data when the required inputs are present. An end-user

Distribution Method	EDGRS Base Tables	Comments
	OMGranule, OMFile, users	subscription is identified in the system as a record with no order id but has a usertype indicating End-User (dist table)
Data Pools	dpgranule, users	The data pool supports the end-user distribution
Anonymous FTP Transfers	SCRS_ftp, SCRS_ftpusrs, SCRS_ftp_fileid, SCRS_ftp_hostid, SCRS_ftpusrs_host	These are ftp-pulls from the non-ECS systems.
Web Downloads	SCRS_wdd, SCRS_wwwusrs, SCRS_wdd_fileid	Web Downloads are products (usually images) posted on a web site. When the users click on the image to download it to their machine, it is considered a web delivery or web download. Icons and other images that are used for window dressing or control are not products.
Web Accesses	SCRS_www, SCRS_wwwusrs	Web accesses is a count of the number of accesses by a given user on a given day.
Offline Inquiry Accesses	SCRS_inq, SCRS_inqurs	Offline inquiries include requests such as emails, walk-in requests, phone requests, and other manually provided services.

### 5.5.1.1 The Dist Table

The dist table is used to merge information from the various transactions pertaining to orders and subscriptions. It contains both end-user transactions as well as those used for production processing and QA/test purposes. This table resolves many issues to provide a single source of distribution metrics for orders and subscriptions. These issues include the following:

- Determining the proper state of a transaction. An order can be successfully staged but cancelled before delivery. This transaction state should be ‘cancelled’. For orders, the final state comes from the ecacrqst or OMRequest tables.
- Resolving records that have ‘MULTIPLE’ as the datatype to their actual datatype components. This creates multiple records for the same request. The datatype field that reflects this resolution is the ‘calcesdt’ field.
- For PDS records, a complicated algorithm is used to lookup proper field information. See the Section 5.5.1.4 Handling PDS Transactions for more information on how this is done.

When using the dist table, the following assumptions apply:

1. An order can contain multiple requests.
2. A request cannot contain more than one ESDT unless its datatype is ‘MULTIPLE’.
3. Requests whose datatype is not ‘MULTIPLE’ can be filled on different days. An order that contains such requests would then span more than one day.

4. A request is considered an end-user request unless it is otherwise mapped in the subscr table.
5. Only requests with a state of 'Shipped' are considered as completed successful requests.
6. The affiliation of a user is determined from the email address and last name information, when available. This information is obtained from the users table. Only the part of the email address after the '@' sign is used for this purpose. Email addresses are known to be an inaccurate indication of the user domain (e.g., .com vs. .gov vs. .edu) since a government user may use a .com address. However, this is the only measure we have available for this purpose. Users are not required to register.
7. Data types (ESDTs) are uniquely mapped to mission, instrument, level, and discipline in the datatype table.
8. Orders that span more than one day can cause multiple counting of orders in ad-hoc summary reports generated from daily summary tables, i.e. from tables based on daily sums. This over-counting is corrected when the ad-hoc reports are generated.
9. Requests may have no associated orderids.
10. For gsfcv0 data, different requests processed may have the same requestid. These correspond to different requests for the same data.

## Metrics

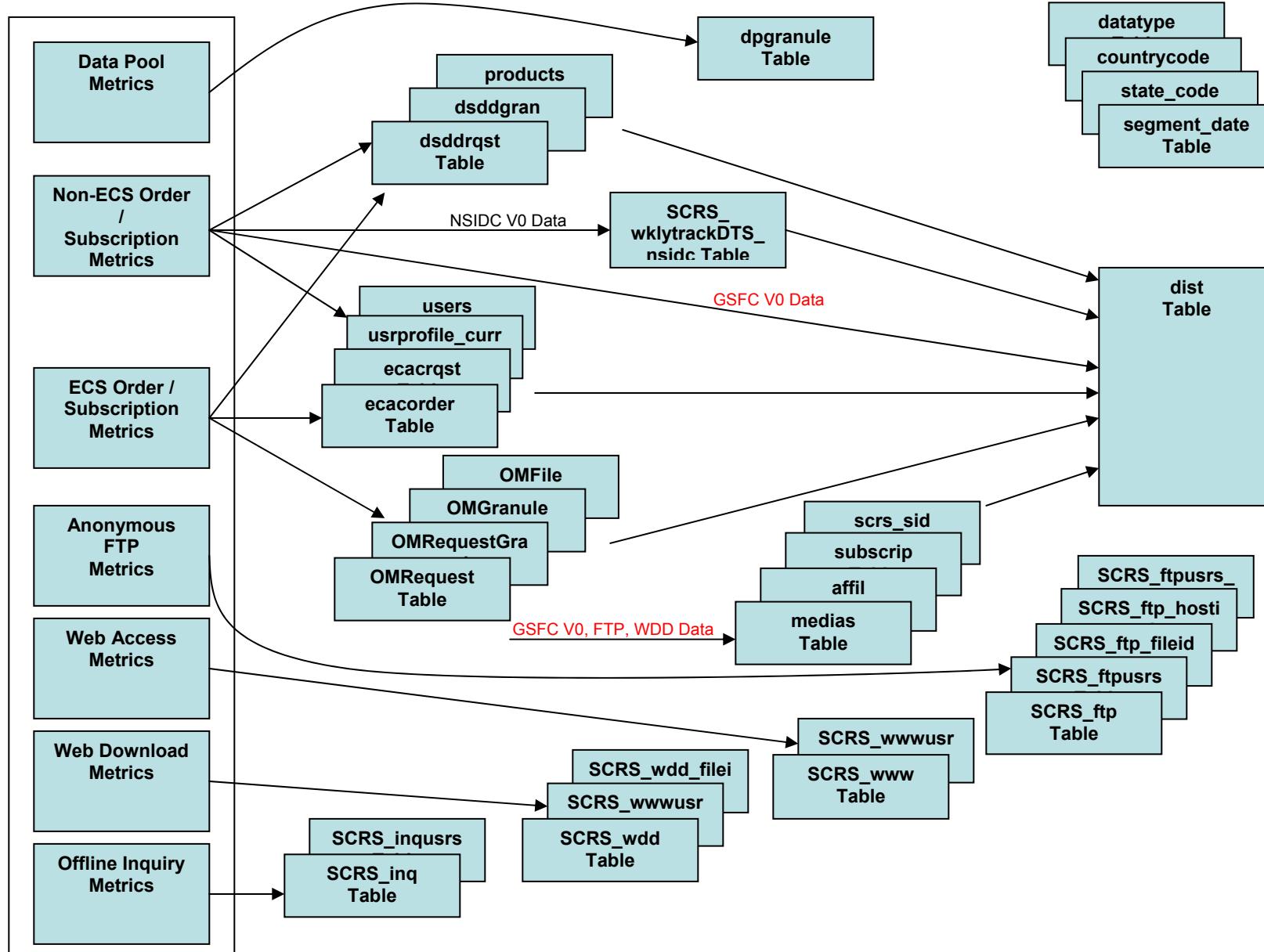


Figure 5-4 EDGRS Distribution Subsystem – Base Table Overview

**Table 5-15 Description of Tables in Dist Subsystem**

Table ID	EDGRS Table Name	Primary Key	EDGRS Table Description	Comments
1	dist	provider, requested, calcesdt, starttime	The dist table is the primary source of metrics on data distributed from EOSDIS sites. The dist table keeps a list of each request distributed from the ECS and non-ECS DAACs. Supporting tables are loaded nightly from a set of flat files sent by the DAACs representing recent transactions. These supporting tables are then used to create the dist table while resolving multiple issues with data status and identification.	Note: starttime is used in the key only for data base segmentation, not for uniquely finding records. To find a distinct record, only the first three fields of the key are required. Note that not all fields are always populated. See the field definitions for more information.
2	datatype	datatypeid  OR  esdt_id	The datatype table is used to uniquely map the ESDT names to mission, instrument, level, and discipline for reporting purposes.	Datatypeid is the ESDT name used by the DAAC. Esdt_id is an internally coded value used by EDGRS. Either field may be used to uniquely identify or link a datatype. Note that the only reliably completed fields in this table are esdt_id, datatypeid, descriptio (description text), edginstrum (the instrument), mission, level, and discipline. Other fields were used in initial data gathering efforts and have no guarantee of useful information.  Note. This table is maintained manually. The datatype table is joined when summary tables are created or updated. The mission, instrument etc information is not kept in the base tables.
3	affil	code	The affil table defines the categories of affiliation. Both US and foreign affiliations are defined for the calculated affiliation (based on email address). User specified affiliations (code > 50) are also defined.	This table is maintained manually.
4	subscrip	provider, userid	The subscrip table is used to identify a subscription's purpose and categorization. The purpose is represented in the required category field as 'End-User', 'Production', or 'QA/Test'. Further categorization is optionally provided in the rollup and userspecified fields.	Userids not in this table are assumed to be 'End-User' transactions.  Note. This table is maintained manually.

<b>Table ID</b>	<b>EDGRS Table Name</b>	<b>Primary Key</b>	<b>EDGRS Table Description</b>	<b>Comments</b>
5	medias	media_id OR media_txt	The medias table is used to normalize the various textual media names into a common media name.	When joining with media from the dist table, the media_id field is used as the key. When loading the dist table, the media_text field is used to match text provided in the input flat files.
6	usrprofile_curr	provider, userid	The usrprofile_curr table is used to record registered users. It contains information about the user at the time of registration.	
7	users	edrsid OR emailaddr, lastname	The users table is used to record distinct users in EDGRS. It can be populated from the usrprofile data or from order data. This table is updated whenever information changes.	This table is updated automatically from the usrprofile table or from new orders. Information in the table such as shipping addresses or other non-key fields can be updated. The edrsid field is used when joining with dist for reporting purposes. The emailaddr+lastname key is used when loading or updating this table.
8	products	prodname	The products table is used to normalize the product name	
9	dsddgran	Provider, requestid, granuleid, starttime		
10	dsdrqst	Provider, requestid, starttime		
11	ftp	None		
12	ftphost	None		
13	ftppullhos	None		
14	ftppushdes	None		
15	dpgranule	Provider, dbid, accessType ,fileType, accesstime		
16	ecacrqst	Provider, orderid, requestid		
17	ecacorder	Provider,		

<b>Table ID</b>	<b>EDGRS Table Name</b>	<b>Primary Key</b>	<b>EDGRS Table Description</b>	<b>Comments</b>
		orderid		
18	OMFile	Provider, fileid		
19	OMGranule	Provider, granid		
20	OMRequest Granule	Provider, requestid, granid_no, granid		
21	OMRequest	Provider, requestid		
22	OMMeidaType	Provider, mediatypeid		
23	OMExplanation	Provider, explanation code		
24	OMStatus	Provider, statuscode		
25	OMConfigFTPPushDest	Provider, destinationid		
26	SCRS_inq	Daac, sid, inq_dt, inq_tm, cust_id		
27	SCRS_inquiries	Lname, addr		
28	scrs_sid	Source_name, source_txt		
29	SCRS_ftp	Daac, sid, ftp_dt, ftp_tm, fileid, hosted, cust_id,		

<b>Table ID</b>	<b>EDGRS Table Name</b>	<b>Primary Key</b>	<b>EDGRS Table Description</b>	<b>Comments</b>
		cust_host, datatype		
30	SCRS_ftpusrs	addr		
31	SCRS_ftp_fileid	filename		
32	SCRS_ftp_hostid	hostname		
33	SCRS_ftpusrs_host	addr, hostid		
34	SCRS_www	daac, sid, www_dt, cust_id		
35	SCRS_www_usrs	host		
36	SCRS_wdd	daac, sid, www_dt, www_tm, fileid, cust_id, datatype		
37	SCRS_wdd_fileid	filename		
38	countrycode	None		
39	State_code	None		
40	Segment_date	None		

Add descriptions of medias, products, ftp, ecac, dsdd, OM. SCRS and other tables (TBS)

### **5.5.1.2 Summary Tables for Reporting Metrics on ECS Data**

Text TBS

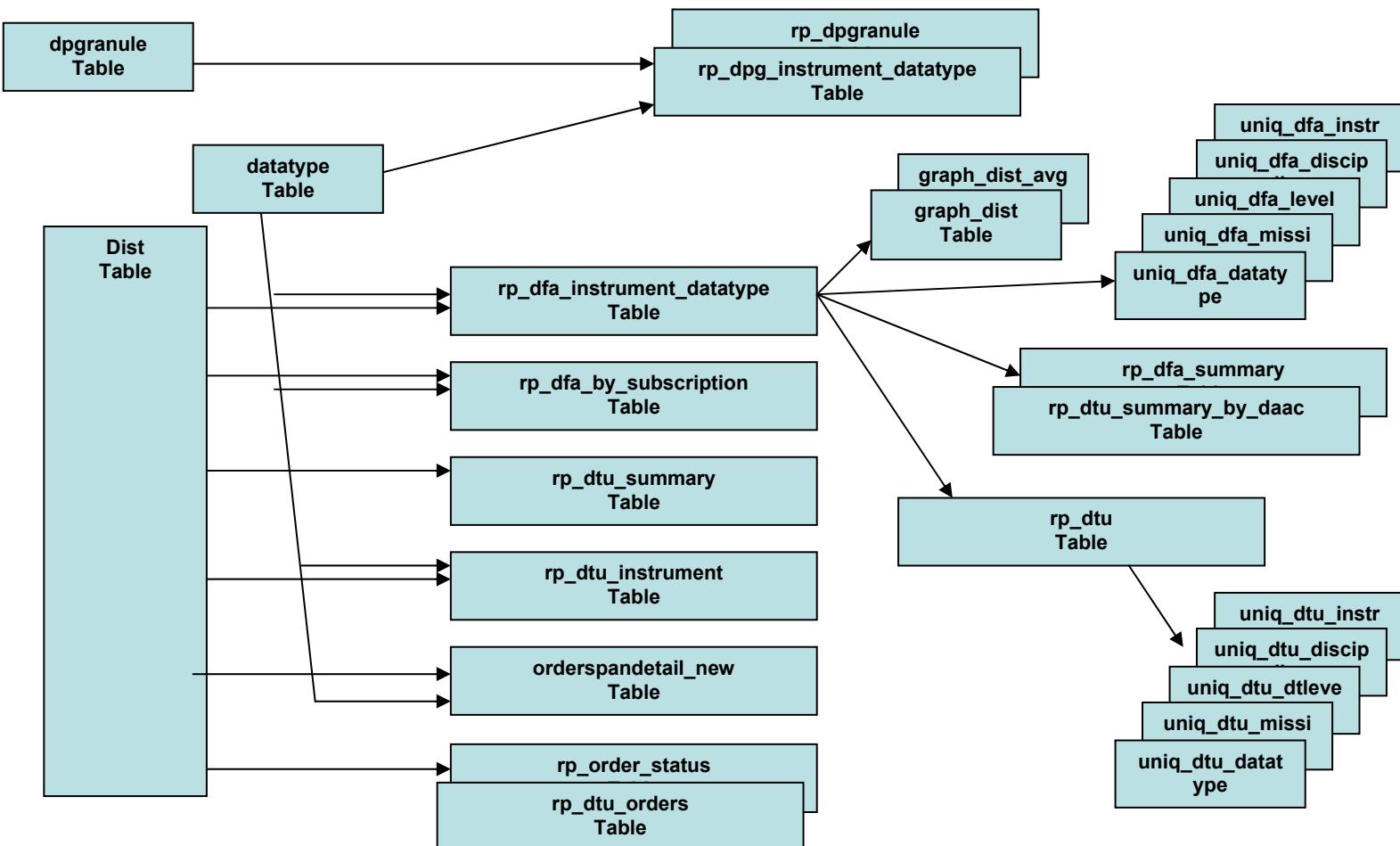


Figure 5-5 EDGRS Distribution Subsystem Summary Table Overview – Order/Subscription Data

*Table 5-16 Description of dfa and dtu Summary Tables in Distribution Subsystem*

Table ID	EDGRS Table Name	Primary Key	EDGRS Table Description	Comments
			(TBS)	

#### 5.5.1.3 Summary Tables for Reporting on non-ECS (Log) Data

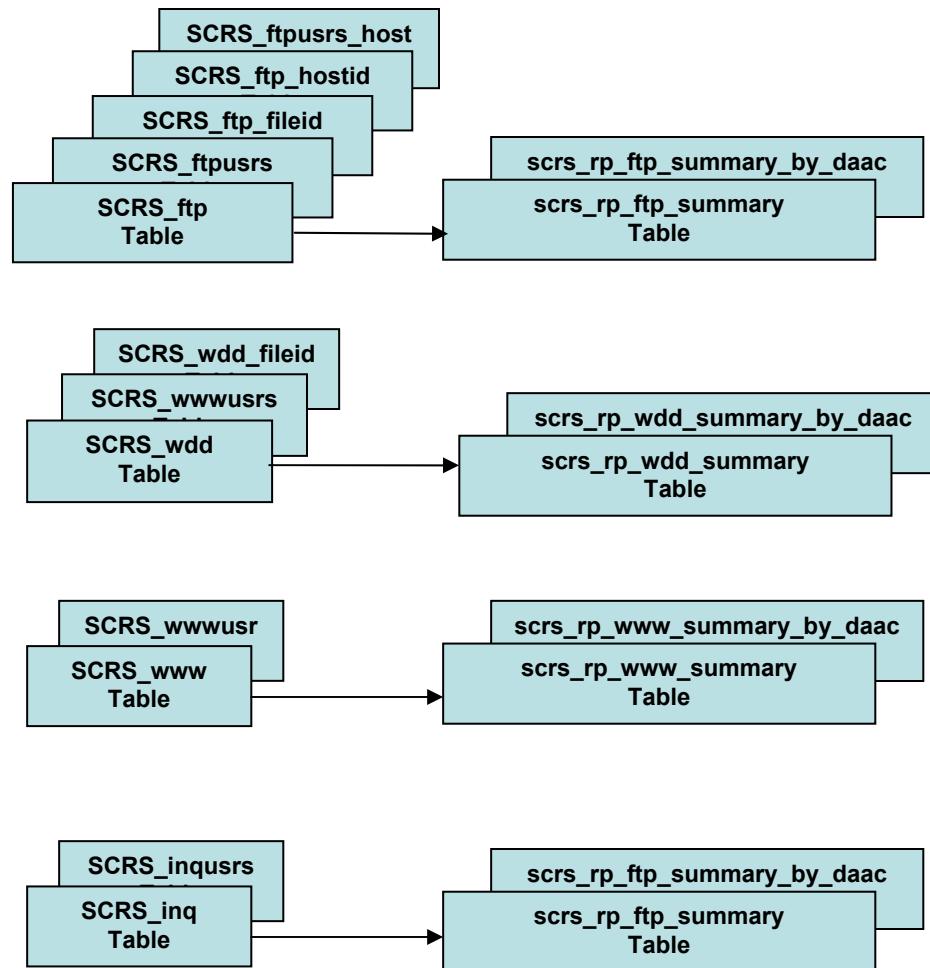


Figure 5-6 EDGRS Distribution Subsystem Summary Table Overview – Log Data

*Table 5-17 Description of SCRS Summary Tables in Distribution Subsystem*

Table ID	EDGRS Table Name	Primary Key	EDGRS Table Description	Comments

Table ID	EDGRS Table Name	Primary Key	EDGRS Table Description	Comments
			(TBS)	

#### 5.5.1.4 Handling PDS transactions

Gathering metrics for PDS transactions in the Storage Management Subsystem of ECS is complicated due to the convoluted way to map the completions of the media generation to the original request. The method to track the status and granule count/size components of a PDS request is described as follows:

In the ECS, media orders are handled by the PDS system. The original request is logged in the EcAcOrder/EcAcRequest environment. Then a new request is entered into the Storage Management Subsystem. Eventually it gets marked 'Shipped' in the DsDdRequest table. PDS records have a '\$PDS' as its EcsUserId in the DsDdRequest table and its MediaType is 'FtpPush' in the DsDdParameterList table. The UserString field in the DsDdParameter List table is composed of:

**PDS:<ECS requestid>:<granule number in PDS>:<number of times sent to ECS>**

To check to see if this granule has already been counted, the GranuleId in the DsDdGranule table and the ECS request Id that is embedded in the UserString field are checked against a list of all granules successfully shipped. If this requestId/ GranuleId combination is not found, then the numBytes field is incremented by the value in the GranuleSize field in the DsDdGranule table and numGranule field is incremented by one in the EcAcRequest table. This allows reporting on a request level basis.

#### 5.5.1.5 Order Management System (OMS) - TBS

##### 5.5.2 Building the EDGRS dist Table

###### 5.5.2.1 FoxPro Processing (TBS)

### **5.5.2.2 SQL Server Processing**

The dist table in SQL Server is segmented into multiple tables. This is done for both performance and storage allocation reasons. There are views named ‘view\_dist\_all’, ‘view\_dist\_old’ and ‘view\_dist\_new’ which link the table or parts of it together for queries. The segmentation procedure is described in Item #7 of the ‘HowDoI.doc’ file and should be done periodically, currently every six months.

All segments of the dist table remain online. The older and newer segments are grouped separately solely in order to speed up the generation of the dist summary tables.

The data flow diagrams, procedure/view definitions, table definitions and index definitions for tables used to build the dist table are described below.

Notes:

- All procedures and tables described below are in the EdgrsCurrent database except where indicated otherwise.
- The field definitions for the tables described here appear separately in the document ‘Release 1 EOSDIS Data Gathering and Reporting System (EDGRS) Database Design and Database Schema Specifications’.

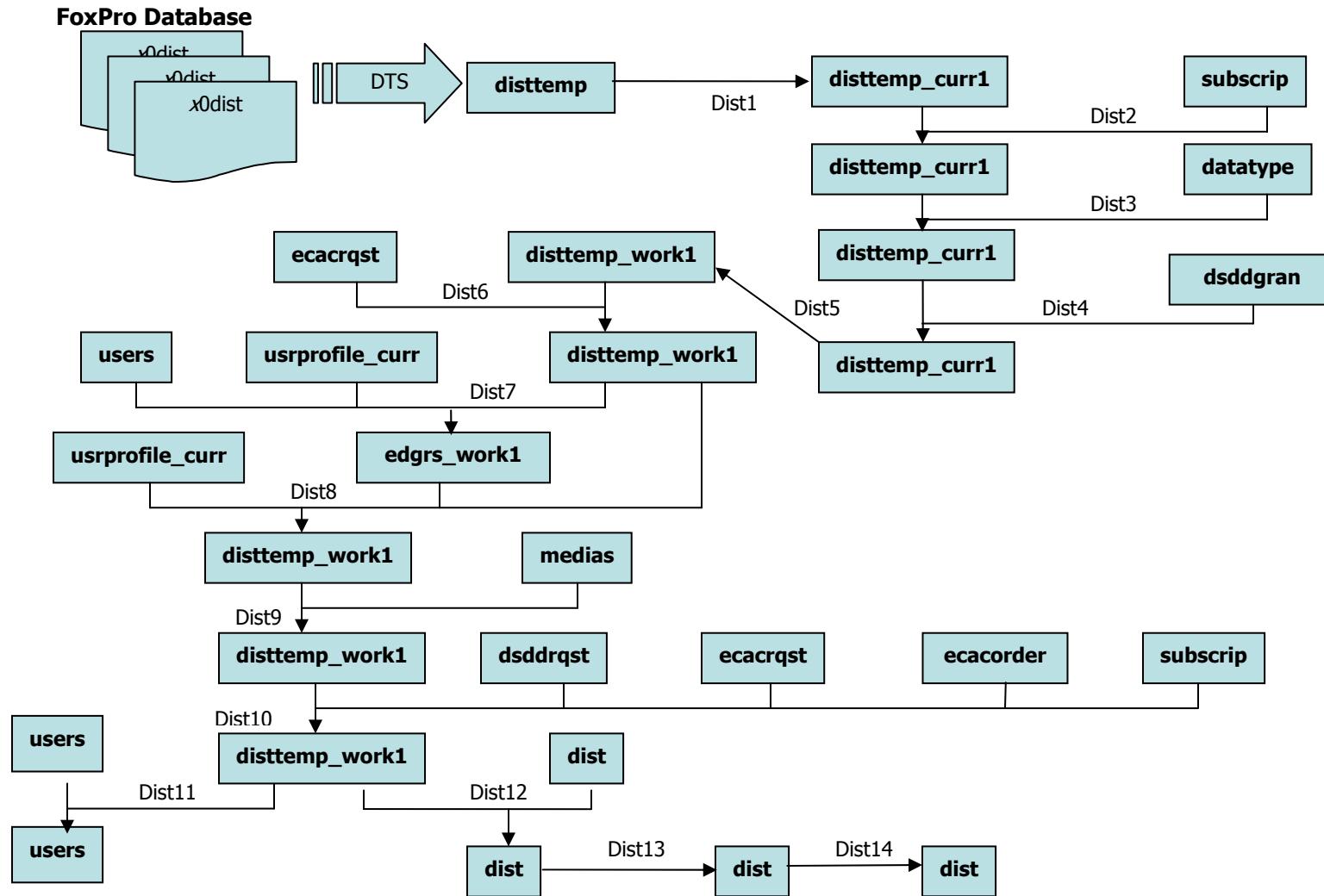


Figure 5-7 Distribution – Dist Ingest DFD

**Table 5-18 Distribution – Dist Ingest DFD Key**

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
Dist1	pro_update_dist_curr1	Update_dist_curr1.sql	Copy all records to disttemp_curr1	
Dist2	pro_update_dist_curr1	Update_dist_curr1.sql	Set usertype in disttemp_curr1	Derive usertype using subscrip table
Dist3	pro_update_dist_curr1	Update_dist_curr1.sql	Set esdt_id in disttemp_curr1	Derive esdt_id using datatype table.
Dist4	pro_update_dist_curr1	Update_dist_curr1.sql	Set bytes in disttemp_curr1	Derive bytes using dsddgran table.
Dist5	pro_update_dist_curr1	Update_dist_curr1.sql	Copy all records to disttemp_work1	Set last 2 fields of disttemp_work1 to null
Dist6	pro_update_dist_curr1	Update_dist_curr1.sql	Set state and calcaffili in disttemp_work1	Derive fields using ecacrqst.
	pro_update_dist_curr1	Update_dist_curr1.sql	Update edgrsid in disttemp_work1 for non-PDS orders	Uses ecacrqst and users
Dist7	pro_update_dist_curr1	Update_dist_curr1.sql	Generates edgrsid_work1 for non-orders	Uses usrprofile_curr table via provider and userid to get at users table
Dist8	pro_update_dist_curr1	Update_dist_curr1.sql	Set edgrsid in disttemp_work1 for non-orders using edgrsid_work1	
Dist9	pro_update_dist_curr1	Update_dist_curr1.sql	Set media_id in disttemp_curr1	Derive media_id using medias table.
Dist10	pro_update_dist_curr1	Update_dist_curr1.sql	Update disttemp_curr1 for PDS records	
Dist11	pro_update_dist_curr1	Update_dist_curr1.sql	Update firstdate in users table	.
	pro_update_dist_curr1	Update_dist_curr1.sql	Update usertype to 3 (Production) for NSIDC user 'Hew Subsetter' having userid of 'ECSGuest'	
Dist12	pro_update_dist_curr1	Update_dist_curr1.sql	Update existing records in dist table using disttemp_work1	
Dist13	pro_update_dist_curr1	Update_dist_curr1.sql	Insert new records into dist table using disttemp_work1	

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
Dist14	pro_update_dist_curr1	Update_dist_curr1.sql	Delete extraneous 'MULTIPLE' records from dist	These are the old PDS records that have been replaced by updates in Dist10

**Table 5-19 Definitions for Tables Used to Ingest Data into dist from FoxPro**

Table ID	EDGRS Table Name	Primary Key	EDGRS Table Description	Comments
1	disttemp*	provider requested calcesdt starttime	staging area for dist data transferred from FoxPro	The structure of this file mimics the structure of the input data. Data is moved into this area and then additional processing is performed before moving the data into the dist table.
2	disttemp_curr1*	provider requested calcesdt starttime	staging area for dist data with the same structure as disttemp.	
3	disttemp_work1	provider requested calcesdt	staging area for dist data with two additional fields	The two additional fields are edgrsid and media_id
4	edgrsid_work1	None	staging area for user information: edgrsid, emailaddr and lastname	
5	EdgrsCurrent.dbo.dist EdgrsDist.dbo.dist<x>	provider requested calcesdt starttime	These tables are the segments of the dist table which contains one record for each granule/product distributed.	dist, the current segment, resides in the EdgrsCurrent database. The other segments currently reside in the EdgrsDist database. At some later date newer segments split off from the current segment may be placed in another database to limit the size of the databases containing dist segments. All dist segments have the same structure but different check constraints on starttime. (See 'Howdol.doc' Item#7) . starttime is part of PK only because of segmentation.

- PK is not needed here. It should be removed to speed up processing

**Table 5-20 Index Definitions for Tables Used to Ingest Data into dist**

Index ID	EDGRS Index Name	Table	EDGRS Index Description	Clust-ered	Unique Index	Comments
1	PK_disttemp	disttemp	Primary Key	Yes	Yes	Not needed
2	PK_disttemp_curr1	disttemp_curr1	Primary Key: provider, requestid, calcesdt, starttime	Yes	Yes	Not needed
3	PK_disttemp_work1	disttemp_work1	Primary Key: provider, requestid, calcesdt	Yes	Yes	
4	PK_dist_2 PK_dist_xx_1 PK_dist_04S1_0	dist EdgrsDist.dbo.distxx_1 EdgrsDist.dbo.dist04S1	Primary Key: provider, requestid, calcesdt, starttime	Yes	Yes	
5	INDX_sortendtime INDX_sortendtime_xx_1 INDX_sortendtime_04S1_0	dist EdgrsDist.dbo.distxx EdgrsDist.dbo.dist04S1	indexed on sortendtime	No	No	For speed

### 5.5.3 Building the EDGRS SummaryTables Based on the dist Table

The dist table is used to generate reports. To speed up report generation, data from the dist table is preprocessed by grouping over date and other fields and the results are stored in various summary tables.

The data flow diagrams, procedure/view definitions, table definitions and index definitions for tables used to build the dist summary tables are described below.

Notes:

- All procedures and tables described below are in the EdgrsCurrent database except where indicated otherwise.
- The field definitions for the tables described here appear separately in the document ‘Release 1 EOSDIS Data Gathering and Reporting System (EDGRS) Database Design and Database Schema Specifications’.

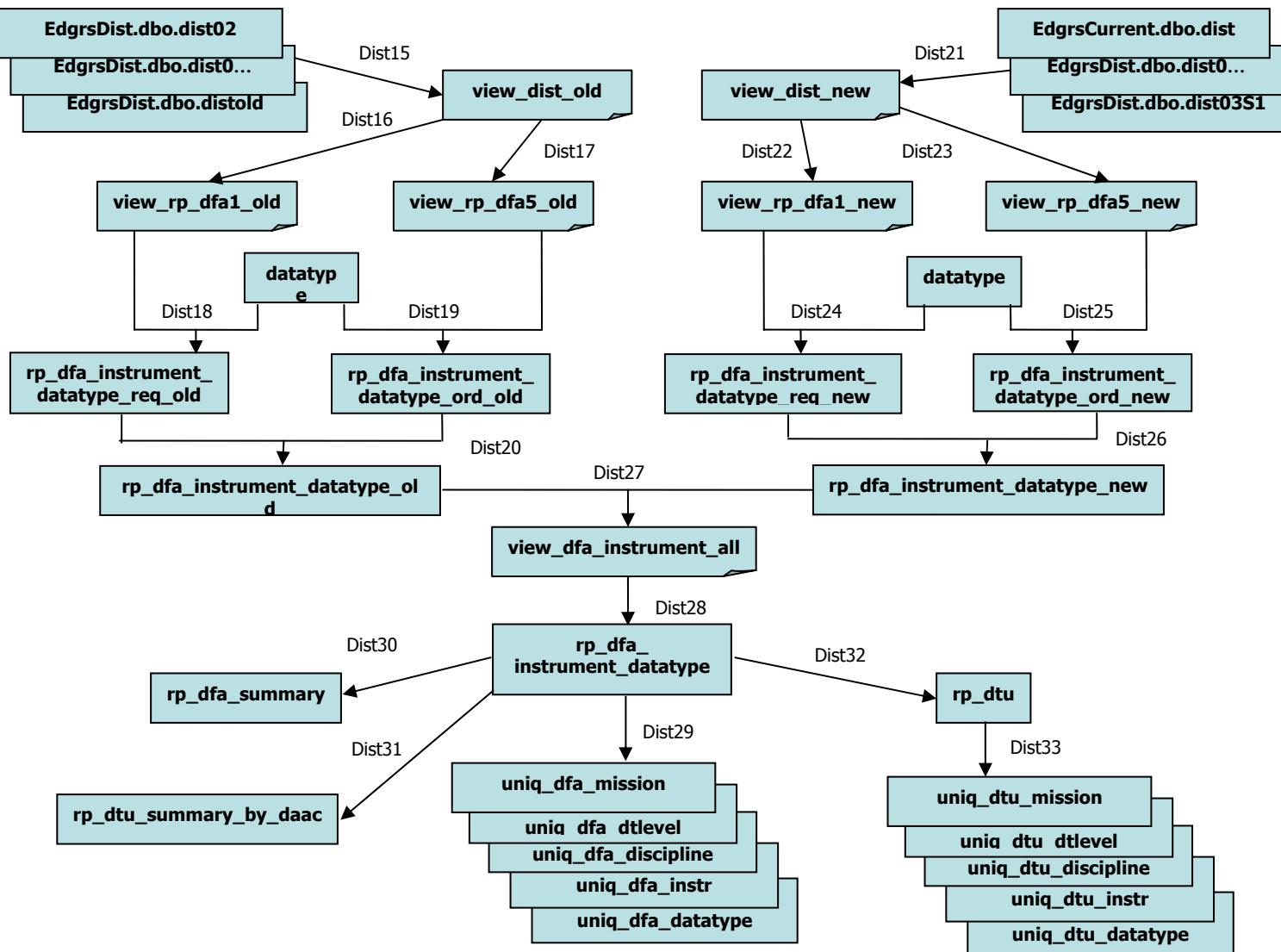


Figure 5-8 Dist Report Generation –from Archive DFD

**Table 5-21 Dist Report Generation –from Archive DFD Key**

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
Dist15	view_dist_old	Create_dist_views.sql	View of dist offline segments	via UNION ALL
Dist16	view_rp_dfa1_old	Create_dist_views.sql	View to get request data from offline segments	State='Shipped'
Dist17	view_rp_dfa5_old	Create_dist_views.sql	View to get order data from offline segments	State='Shipped' and orderid <> ''
Dist18	pro_rp_dfa_instrument_old	Sps0_dist_prod_curr_old.sql	Get user request metrics from offline dist segments using view_rp_dfa1_old	Uses rp_dfa_instrument_datatype_work. Gets datatype fields from datatype table. Sets null values to 'UNKNOWN'.
Dist19	pro_rp_dfa_instrument_old	Sps0_dist_prod_curr_old.sql	Get order metrics from offline dist segments using view_rp_dfa5_old	Uses rp_dfa_instrument_datatype_work. Gets datatype fields from datatype table. Sets null values to 'UNKNOWN'.
Dist20	pro_rp_dfa_instrument_old	Sps0_dist_prod_curr_old.sql	Join request and order metrics to generate rp_dfa_instrument_datatype_old table	Uses rp_dfa_instrument_datatype_work.
Dist21	view_dist_new	Create_dist_views.sql	View of dist online segments	via UNION ALL
Dist22	view_rp_dfa1_new	Create_dist_views.sql	View to get request data from online segments	State='Shipped'
Dist23	view_rp_dfa5_new	Create_dist_views.sql	View to get order data from online segments	State='Shipped' and orderid <> ''
Dist24	pro_rp_dfa_instrument_new	Sps0_dist_prod_curr_new.sql	Get request metrics from online dist segments using view_rp_dfa1_new	Uses rp_dfa_instrument_datatype_work. Gets datatype fields from datatype

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
				table. Sets null values to 'UNKNOWN'.
Dist25	pro_rp_dfa_instrument_new	Spso_dist_prod_curr_new.sql	Get order metrics from online dist segments using view_rp_dfa5_new	Uses rp_dfa_instrument_datatype_work. Gets datatype fields from datatype table. Sets null values to 'UNKNOWN'.
Dist26	pro_rp_dfa_instrument_new	Spso_dist_prod_curr_new.sql	Join request and order metrics to generate rp_dfa_instrument_datatype_new table	Uses rp_dfa_instrument_datatype_work.
Dist27	view_dfa_instrument_all	Create_dist_views.sql	Merge rp_dfa_instrument_datatype_old and rp_dfa_instrument_datatype_new to create view_dfa_instrument_all	via UNION ALL
Dist28	pro_rp_dfa_instrument_all	Spso_dist_prod_curr_all.sql	Build rp_dfa_instrument_datatype table using view_dfa_instrument_all	
Dist29	pro_rp_dfa_instrument_all	Spso_dist_prod_curr_all.sql	Build uniq_dfa... tables from rp_dfa_instrument_datatype table	
Dist30	pro_rp_dfa_summary_all	Spso_dist_prod_curr_all.sql	Build rp_dfa_summary table from rp_dfa_instrument_datatype table	
Dist31	pro_rp_dtu_summary_by_daac_all	Spso_dist_prod_curr_all.sql	Build rp_dtu_summary_by_daac table from rp_dfa_instrument_datatype table	
Dist32	pro_rp_dtu_summary_all	Spso_dist_prod_curr_all.sql	Build rp_dtu table from rp_dfa_instrument_datatype table	
Dist33	pro_rp_dtu_summary_all	Spso_dist_prod_curr_all.sql	Build uniq_dtu... tables from rp_dtu table	

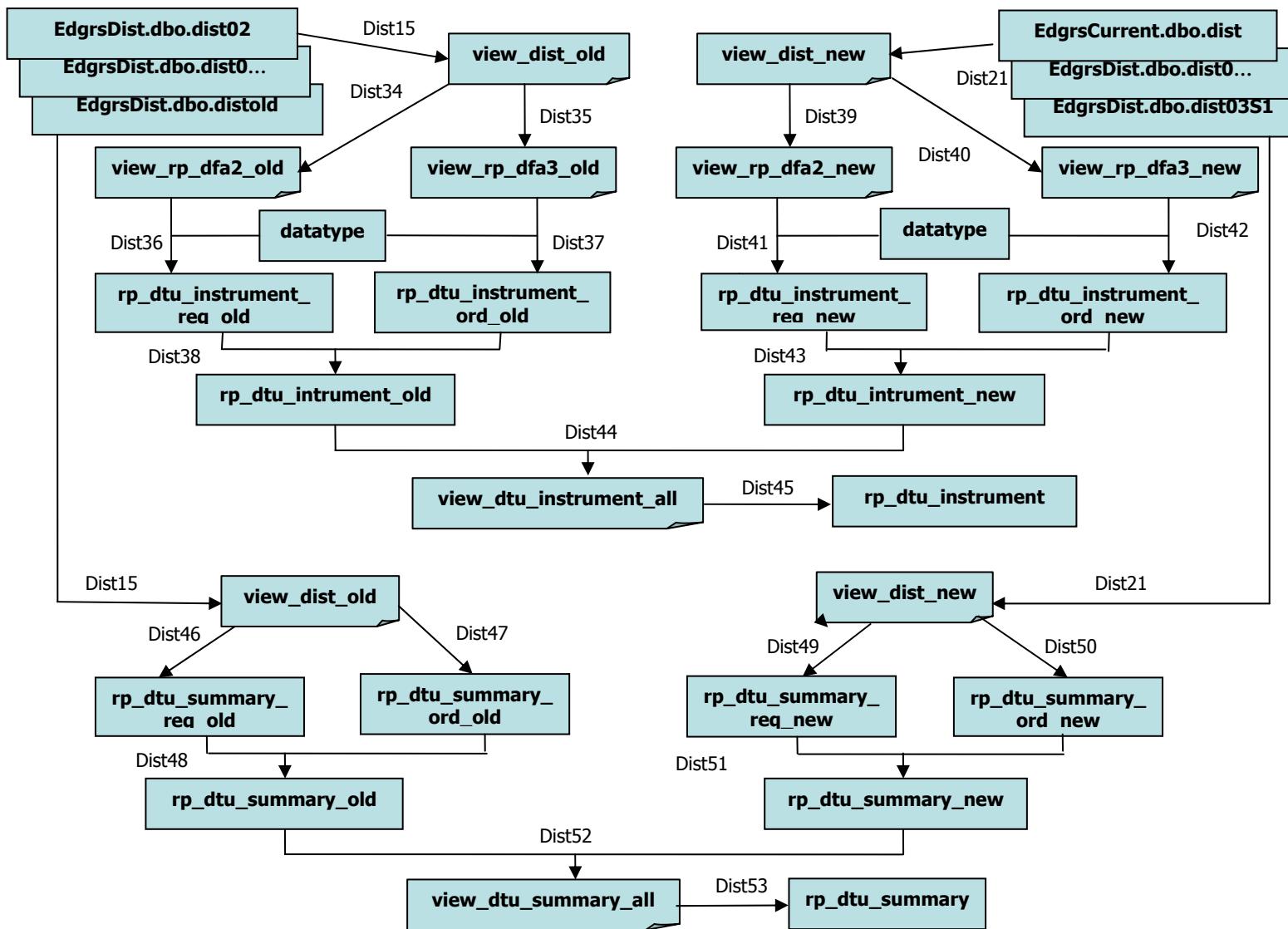


Figure 5-9 Dist Report Generation – to Users DFD

**Table 5-22 Dist Report Generation – to Users DFD Key**

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
Dist34	view_rp_dfa2_old	Create_dist_views.sql	View to get user request data from offline dist segments	State='Shipped' and usertype=2
Dist35	view_rp_dfa3_old	Create_dist_views.sql	View to get user order data from offline dist segments	State='Shipped' and orderid <> '' and usertype=2
Dist36	pro_rp_dtu_instrument_old	Sps0_dist_prod_curr_old.sql	Get request metrics from offline dist segments using view_rp_dfa2_old	Uses rp_dtu_instrument_work. Gets instrument field from datatype table. Sets null values to 'Other'.
Dist37	pro_rp_dtu_instrument_old	Sps0_dist_prod_curr_old.sql	Get order metrics from offline dist segments using view_rp_dfa3_old	Uses rp_dtu_instrument_work. Gets instrument field from datatype table.
Dist38	pro_rp_dtu_instrument_old	Sps0_dist_prod_curr_old.sql	Join request and order metrics to generate rp_dtu_instrument_old table	Uses rp_dtu_instrument_work.
Dist39	view_rp_dfa2_new	Create_dist_views.sql	View to get user request data from online dist segments	State='Shipped' and usertype=2
Dist40	view_rp_dfa3_new	Create_dist_views.sql	View to get user order data from online dist segments	State='Shipped' and orderid <> '' and usertype=2
Dist41	pro_rp_dtu_instrument_new	Sps0_dist_prod_curr_new.sql	Get request metrics from online dist segments using view_rp_dfa2_new	Uses rp_dtu_instrument_work. Gets instrument field from datatype table. Sets null values to 'Other'.
Dist42	pro_rp_dtu_instrument_new	Sps0_dist_prod_curr_new.sql	Get order metrics from online dist segments using view_rp_dfa3_new	Uses rp_dtu_instrument_work. Gets instrument field from datatype table. Sets null values to 'UNKNOWN'.
Dist43	pro_rp_dtu_instrument_new	Sps0_dist_prod_curr_new.sql	Join request and order metrics to generate rp_dtu_instrument_new table	Uses rp_dtu_instrument_work.
Dist44	view_dtu_instrument_all	Create_dist_views.sql	Merge rp_dtu_instrument_old and rp_dtu_instrument_new	via UNION ALL

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
Dist45	pro_rp_dtu_instrument_all	Sps0_dist_prod_curr_all.sql	Join request and order metrics to generate rp_dtu_instrument table	Uses view_dtu_instrument_all.
Dist46	pro_rp_dtu_summary_old	Sps0_dist_prod_curr_old.sql	Get dtu summary request metrics from offline dist segments	Uses rp_dtu_summary_work. State='Shipped' and usertype=2.
Dist47	pro_rp_dtu_summary_old	Sps0_dist_prod_curr_old.sql	Get dtu summary order metrics from offline dist segments	Uses rp_dtu_summary_work. orderid<> "", 'State='Shipped' and usertype=2.
Dist48	pro_rp_dtu_instrument_old	Sps0_dist_prod_curr_all.sql	Join order request and order metrics to build rp_dtu_summary_old table	
Dist49	pro_rp_dtu_summary_new	Sps0_dist_prod_curr_new.sql	Get dtu summary request metrics from online dist segments	Uses rp_dtu_summary_work. State='Shipped' and usertype=2.
Dist50	pro_rp_dtu_summary_new	Sps0_dist_prod_curr_new.sql	Get dtu summary order metrics from online dist segments	Uses rp_dtu_summary_work. orderid<> "", 'State='Shipped' and usertype=2.
Dist51	pro_rp_dtu_instrument_new	Sps0_dist_prod_curr_new.sql	Join order request and order metrics to build rp_dtu_summary_new table	Uses rp_dtu_summary_work.
Dist52	view_dtu_summary_all	Create_dist_views.sql	Merge rp_dtu_summary_old and rp_dtu_summary_new	via UNION ALL
Dist53	pro_rp_dtu_summary_all	Sps0_dist_prod_curr_all.sql	Build rp_dtu_summary table using view_dtu_summary_all	

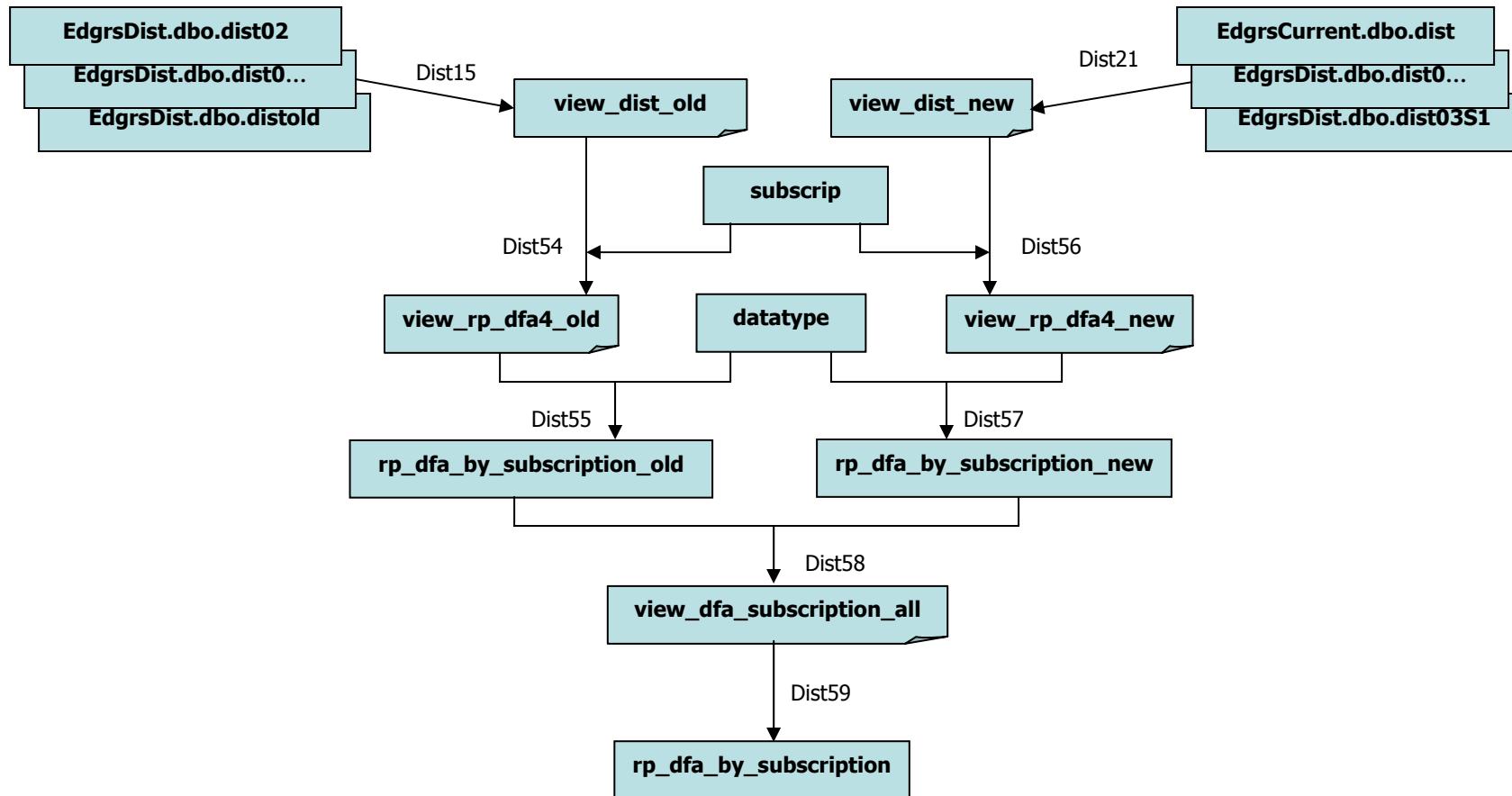


Figure 5-10 Dist Report Generation – Subscriptions DFD

**Table 5-23 Dist Report Generation – Subscriptions DFD Key**

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
Dist54	view_rp_dfa4_old	Create_dist_views.sql	View to get subscription metrics from offline dist segments	Uses subscript table. State='Shipped'.
Dist55	pro_rp_dfa_by_subscription_old	Sps0_dist_prod_curr_old.sql	Get subscription metrics using view_rp_dfa4_old	Gets datatype fields from datatype table. Sets null values to 'UNKNOWN'.
Dist56	view_rp_dfa4_new	Create_dist_views.sql	View to get subscription metrics from online dist segments	Uses subscript table. State='Shipped'.
Dist57	pro_rp_dfa_by_subscription_new	Sps0_dist_prod_curr_new.sql	Get subscription metrics using view_rp_dfa4_new	Gets datatype fields from datatype table. Sets null values to 'UNKNOWN'.
Dist58	view_dfa_subscription_all	Create_dist_views.sql	Merge rp_dfa_by_subscription_old and rp_dfa_by_subscription_new	via UNION ALL
Dist59	pro_rp_dfa_by_subscription_all	Sps0_dist_prod_curr_all.sql	Build rp_dfa_by_subscription table using view_dfa_subscription_all	

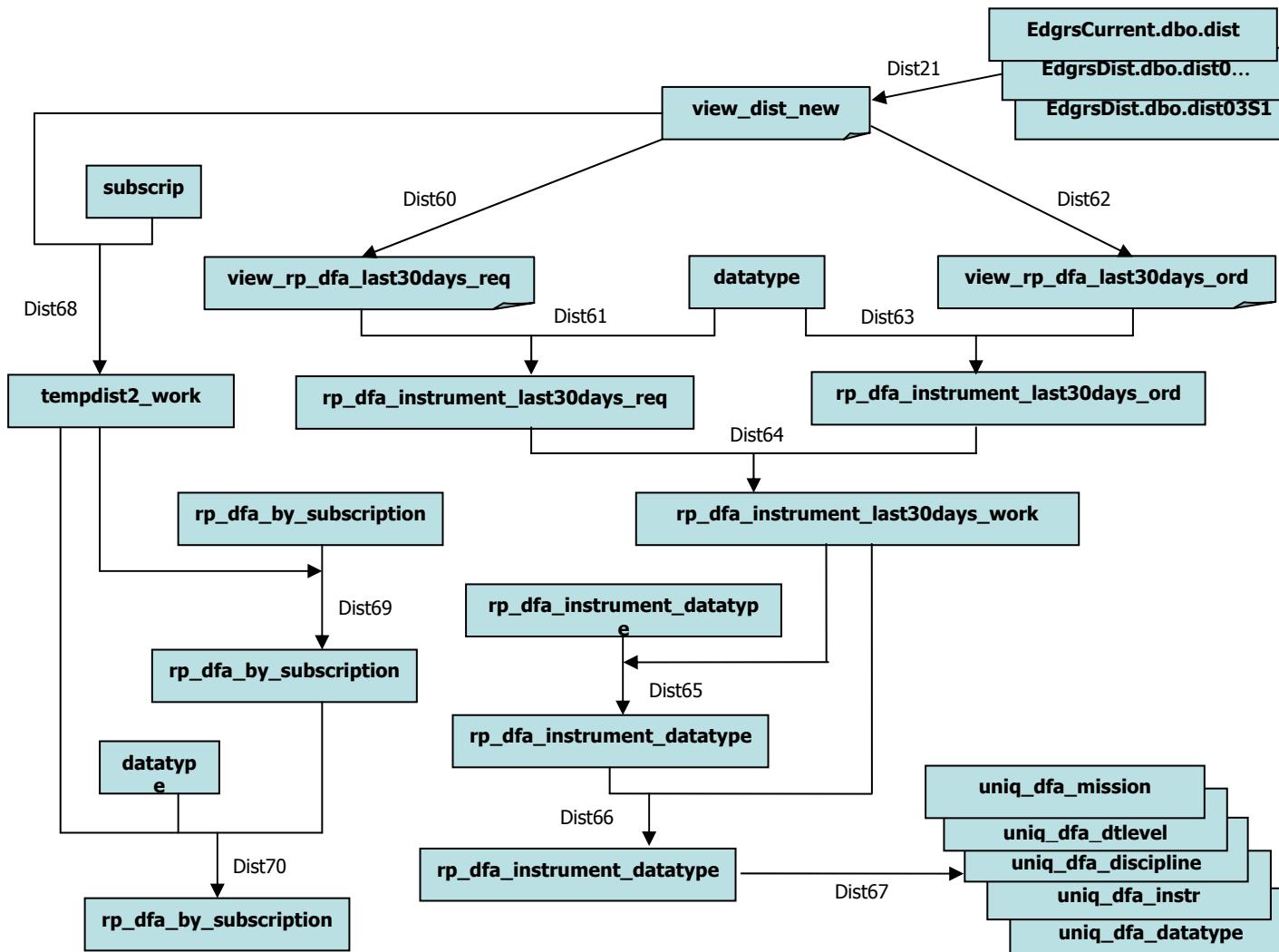


Figure 5-11 Dist – Last 30 Days DFD

**Table 5-24 Dist – Last 30 Days DFD Key**

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
Dist60	view_rp_dfa_last30days_req	Create_dist_views.sql	View of last 30 days of request data from online dist segments	Uses sortendtime. State='Shipped' and sortendtime (endtime) within last 30 days
Dist61	pro_rp_dfa_instrument_last30days_all	Sps0_dist_prod_last30days_all.sql	Build rp_dfa_instrument_last30days_req using view_rp_dfa_last30days_req	Gets datatype fields from datatype table. Sets null values to 'UNKNOWN' or 'OTHER'.
Dist62	view_rp_dfa_last30days_req	Create_dist_views.sql	View of last 30 days of request data from online dist segments	Uses sortendtime. State='Shipped' and orderid <> '' .
Dist63	pro_rp_dfa_instrument_last30days_all	Sps0_dist_prod_last30days_all.sql	Build rp_dfa_instrument_last30days_ord using view_rp_dfa_last30days_ord	Gets datatype fields from datatype table. Sets null values to 'UNKNOWN' or 'OTHER'.
Dist64	pro_rp_dfa_instrument_last30days_all	Sps0_dist_prod_last30days_all.sql	Join request and order metrics to generate table rp_dfa_instrument_last30days_work	
Dist65	pro_rp_dfa_instrument_last30days_all	Sps0_dist_prod_last30days_all.sql	Delete records from rp_dfa_instrument_datatype table that are in rp_dfa_instrument_last30days_work	
Dist66*	pro_rp_dfa_instrument_last30days_all	Sps0_dist_prod_last30days_all.sql	Insert records from rp_dfa_instrument_last30days_work	

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
			into rp_dfa_instrument_datatype table	
Dist67	pro_rp_dfa_instrument_last30days_all	Sps0_dist_prod_last30days_all.sql	Build uniq_dfa... tables from rp_dfa_instrument_datatype table	
Dist68	pro_rp_dfa_by_subscription_last30days_all	Create_dist_views.sql	Get last 30 days of subscription data from online dist segments and store in tempdist2_work table	State='Shipped' and sortendtime(endtime) within last 30 days
Dist69	pro_rp_dfa_instrument_new	Sps0_dist_prod_last30days_all.sql	Delete last 30 days of data from rp_dfa_by_subscription table	
Dist70*	pro_rp_dfa_instrument_new	Sps0_dist_prod_last30days_all.sql	Insert records from tempdist2_work into rp_dfa_by_subscription table	

\* Uses unneeded EXISTS clause

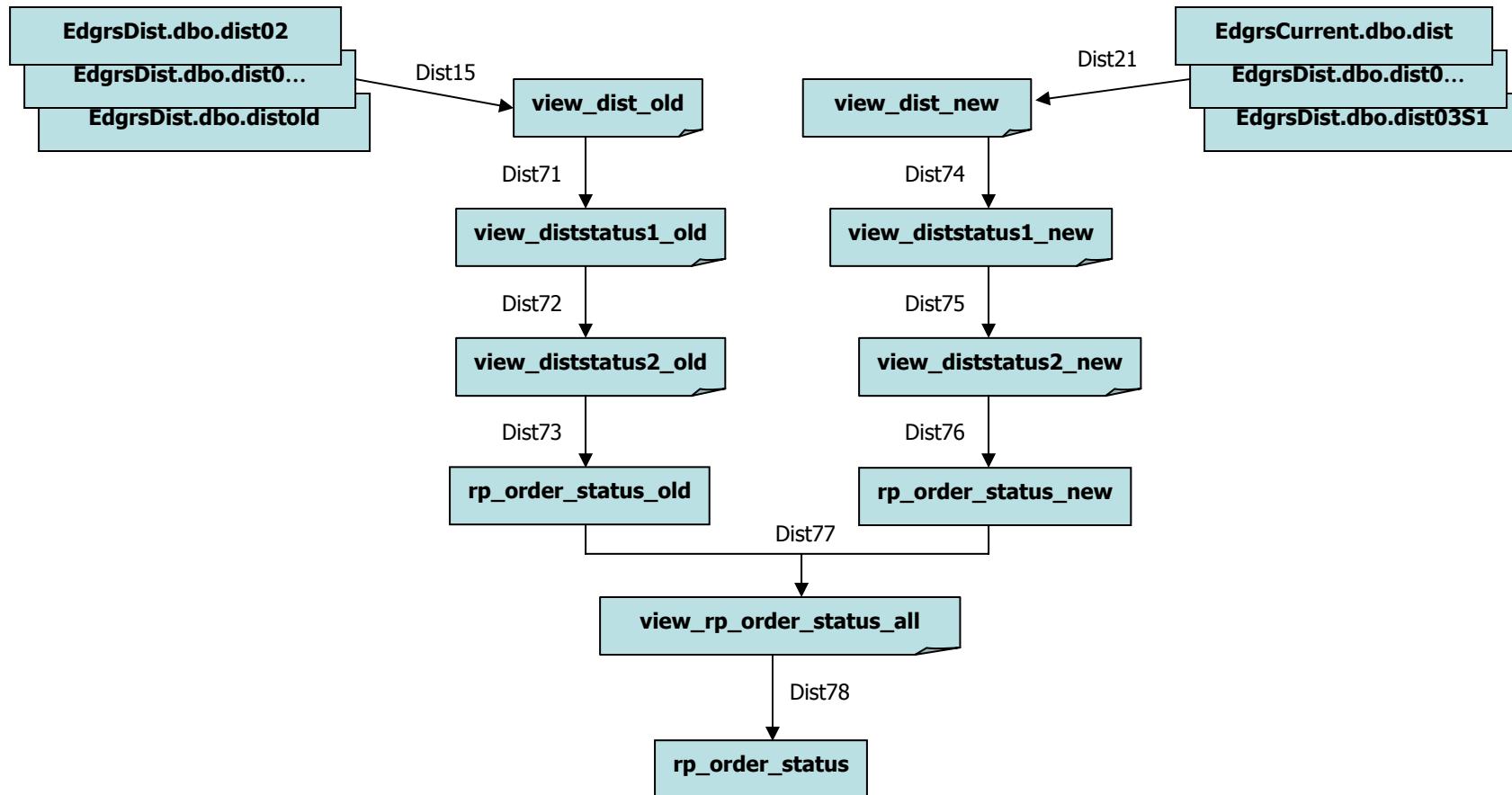


Figure 5-12 Dist Report Generation – Order Status DFD

**Table 5-25 Dist Report Generation – Order Status DFD Key**

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
Dist71	view_diststatus1_old	Create_dist_views.sql	Get status/state information from offline dist segments	Converts endtime to date
Dist72	view_diststatus2_old	Create_dist_views.sql	Break down counts for different states	Uses view_diststatus1_old.
Dist73	pro_rp_order_status_old	Rp_order_status_old.sql	Build rp_order_status_old table using view_diststatus2_old	
Dist74	view_diststatus1_new	Create_dist_views.sql	Get status information from online dist segments	Converts endtime to date
Dist75	view_diststatus2_new	Create_dist_views.sql	Break down counts for different states	Uses view_diststatus1_new.
Dist76	pro_rp_order_status_new	Rp_order_status_new.sql	Build rp_order_status_new table using view_diststatus2_new	
Dist77	view_rp_order_status_all	Create_dist_views.sql	Merge rp_order_status_old and rp_order_stastus_new	via UNION ALL
Dist78	pro_rp_order_status_all	Rp_order_status_all.sql	Build rp_order_status table using view_rp_order_status_all	

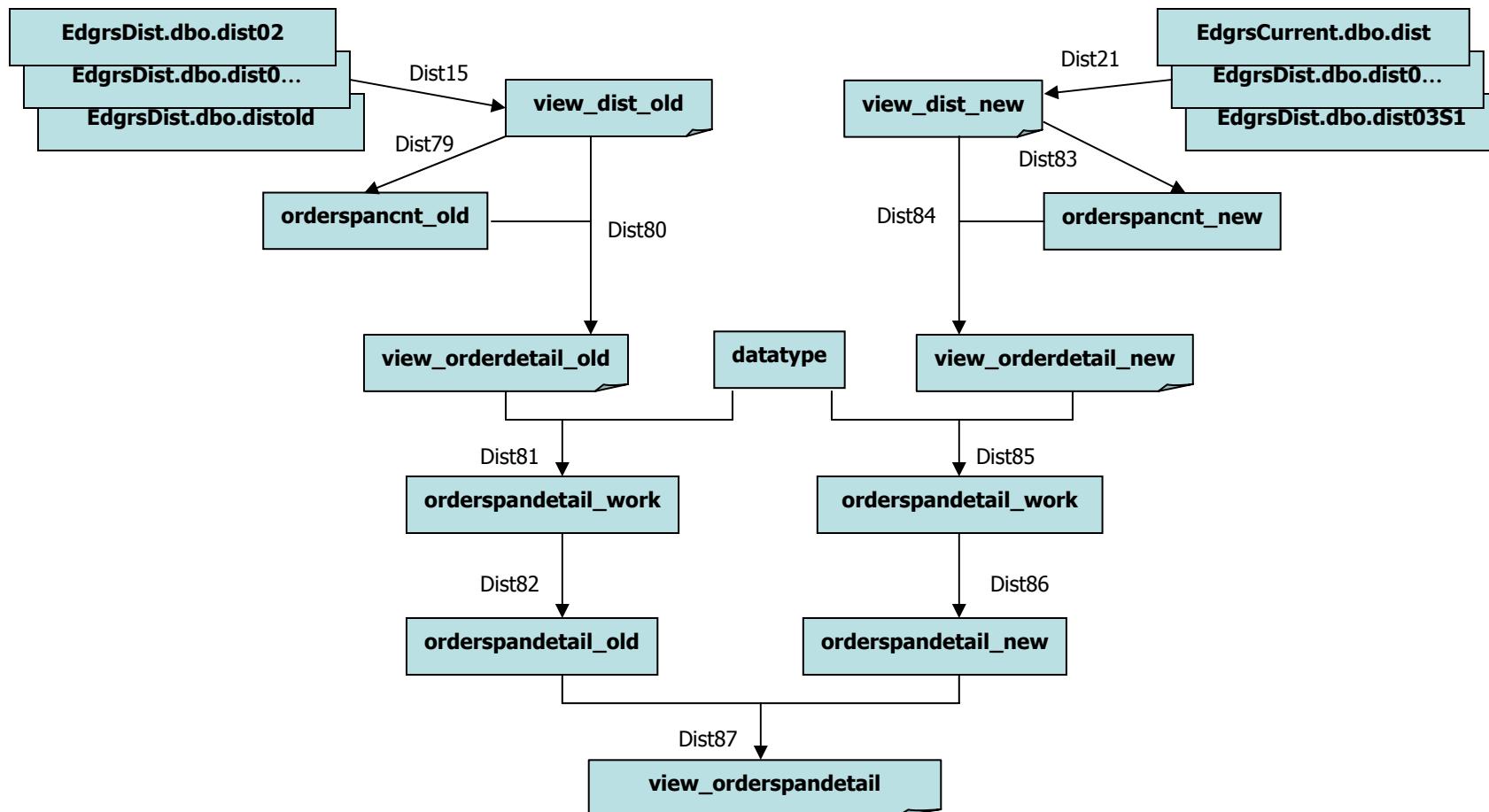


Figure 5-13 Dist Report Generation – Orders Spanning Days DFD

**Table 5-26 Dist Report Generation – Orders Spanning Days DFD Key**

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
Dist79	pro_orderspancnt_old	Sps0_dist_prod_curr_old.sql	Identify orders that span multiple days from offline dist segments	Orderid not "", usertype=2, state='Shipped' and provider not 'GSFCV0'
Dist80	view_orderdetail_old	Create_dist_views.sql	Get detail records for orders spanning days from offline dist segments	
Dist81	pro_orderspandetail_old	Sps0_dist_prod_curr_old.sql	Build orderspandetail_work table using view_orderdetail_old and datatype	
Dist82	pro_orderspandetail_old	Sps0_dist_prod_curr_old.sql	Move offline detail records to orderspandetail_old table	
Dist83	pro_orderspancnt_new	Sps0_dist_prod_curr_new.sql	Identify orders that span multiple days from online dist segments	Orderid not "", usertype=2, state='Shipped' and provider not 'GSFCV0'
Dist84	view_orderdetail_new	Create_dist_views.sql	Get detail records for orders spanning days from online dist segments	
Dist85	pro_orderspandetail_new	Sps0_dist_prod_curr_new.sql	Build orderspandetail_work table using view_orderdetail_new	
Dist86	pro_orderspandetail_new	Sps0_dist_prod_curr_new.sql	Move online detail records to orderspandetail_new table	
Dist87	view_orderspandetail	Create_dist_views.sql	Access detail records for orders that span multiple days	Used to make corrections for order counts in generating dist reports

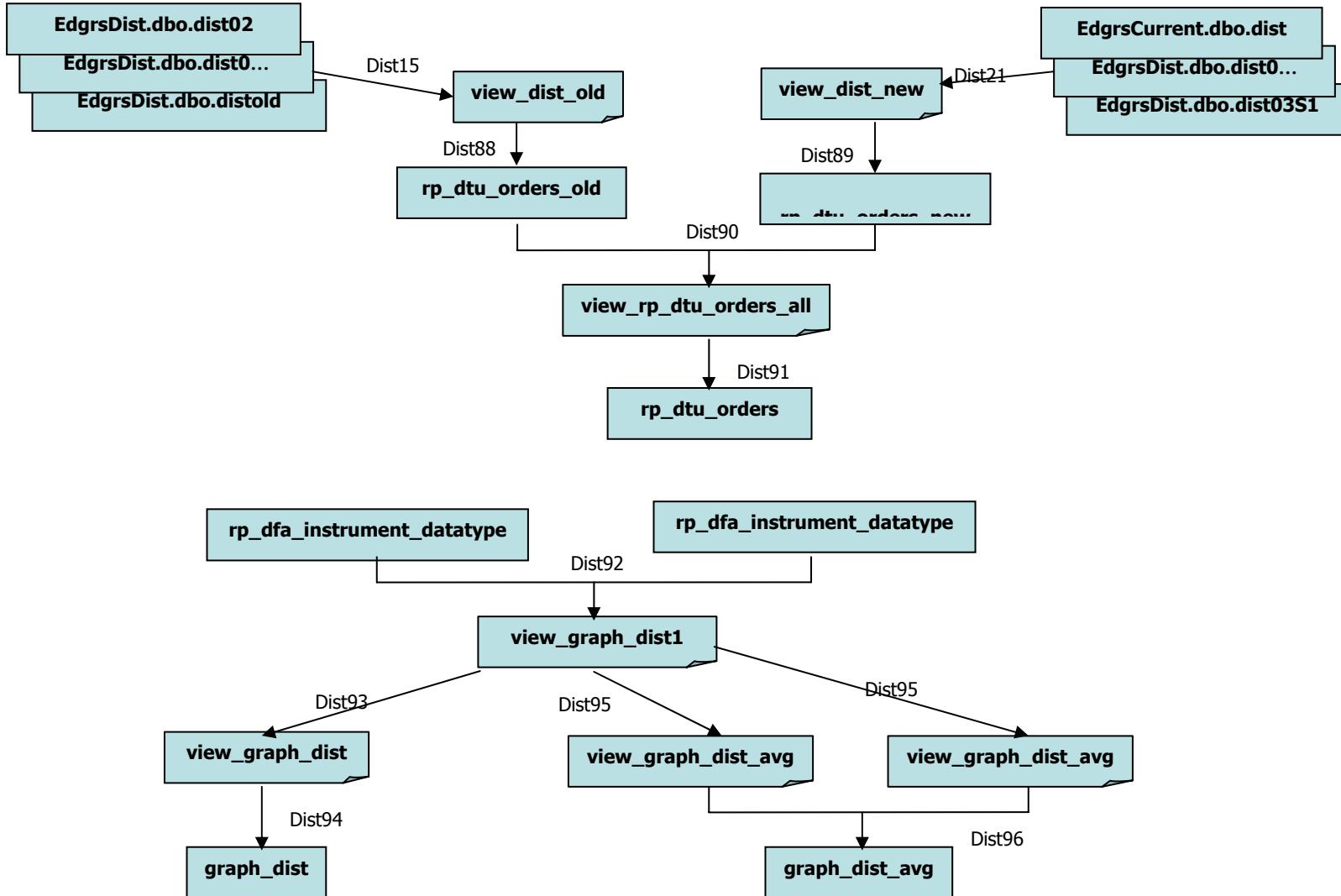


Figure 5-14 Dist Report Generation – Order Counts DFD

**Table 5-27 Dist Report Generation – Order Counts DFD Key**

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
Dist88	pro_rp_dtu_orders_old	Graph_Current/Graph_sps0_dist_old.sql	Get order counts from offline dist segments into rp_dtu_orders_old	Orderid not "", usertype=2 and state='Shipped'
Dist89	pro_rp_dtu_orders_new	Graph_sps0_dist_new.sql	Get order counts from online dist segments into rp_dtu_orders_new	Orderid not "", usertype=2 and state='Shipped'
Dist90	view_rp_dtu_orders_all	Create_dist_views.sql	Merge rp_dtu_orders_old and rp_dtu_orders_new	Via UNION ALL
Dist91	pro_rp_dtu_orders_all	Graph_sps0_dist_all.sql	Build rp_dtu_orders table using view_rp_dtu_orders_all	
Dist92	view_graph_dist1	Create_dist_views.sql	Get record and volume counts for all records and for user records only	Via UNION ALL Uses first char of dtlevel
Dis93	view_graph_dist	Create_dist_views.sql	Merge two types of records from view_graph_dist1 into single record	Sums over discipline
Dist94	pro_graph_dist	Graph_sps0_dist_all.sql	Build graph_dist table using view_graph_dist	
Dist95	view_graph_dist_avg	Create_dist_views.sql	Merge two types of records from view_graph_dist1 into single record	
Dist96	pro_graph_dist_avg	Graph_sps0_dist_all.sql	Build graph_dist_avg table using view_graph_dist_avg	

**Table 5-28 Definitions for Tables Used to Build Summary Tables for dist**

Table ID	EDGRS Table Name	Primary Key	EDGRS Table Description	Comments
1	EdgrsCurrent.dbo.dist EdgrsDist.dbo.dist<xx>	provider requestid calcesdt starttime	These tables are the segments of the dist table which contain one record for each granule/product distributed.	dist, the current segment, resides in the EdgrsCurrent database. The other segments currently reside in the EdgrsDist database. At some later date newer segments split off from the current segment may be placed in another database to limit the size of the databases containing dist segments. All dist segments have the same structure but different check constraints on starttime. (See 'Howdol.doc' Item#7) . starttime is part of PK only because of segmentation.
2	rp_dfa_instrument_datatype		Summary table of daily metrics at the datatype level for all data distributed from archives (orders/subscriptions)	Based on data in segmented dist table. Grouped by provider, insertime, instrument and other parameters
3	rp_dfa_instrument_datatype_(req,new)_old rp_dfa_instrument_datatype_old		Work tables used to generate portion of rp_dfa_instrument_datatype table that comes from offline dist segments	
4	rp_dfa_instrument_datatype_(req,ord)_new rp_dfa_instrument_datatype_new		Work tables used to generate portion of rp_dfa_instrument_datatype table that comes from online dist segments	
5	rp_dfa_summary		Upper level summary table of daily metrics grouped only by provider, insertime and instrument	Based on data in rp_dfa_instrument_datatype table. Values displayed horizontally by daac
6	rp_dtu_summary_by_daac		Upper level summary table of daily metrics grouped only by provider, insertime and instrument and distributed to users	Based on data in rp_dfa_instrument_datatype table. Values displayed horizontally by daac. Usertype=2.
7	rp_dtu		Summary table of daily metrics at the datatype level for data distributed to users only	Based on data in rp_dfa_instrument_datatype table. Usertype=2.
8	uniq_dfa_xxx		Table containing distinct values of parameter xxx occurring in table	xxx = mission, dtlevel, discipline, instr or datatype

<b>Table ID</b>	<b>EDGRS Table Name</b>	<b>Primary Key</b>	<b>EDGRS Table Description</b>	<b>Comments</b>
			rp_dfa_instrument_datatype	
9	uniq_dtu_xxx		Table containing distinct values of parameter xxx occurring in table rp_dtu	xxx = mission, dtlevel, discipline, instr or datatype
10	rp_dtu_instrument		Upper level summary table of daily metrics grouped only by provider, insertime and instrument	Based on data in segmented dist table. Values displayed vertically by daac State='Shipped' and usertype=2.
11	rp_dtu_instrument_req_old rp_dtu_instrument_ord_old rp_dtu_instrument_old		Work tables used to generate portion of rp_dtu_instrument table that comes from offline dist segments	
12	rp_dtu_instrument_req_new rp_dtu_instrument_ord_new rp_dtu_instrument_new		Work tables used to generate portion of rp_dtu_instrument table that comes from online dist segments	
13	rp_dtu_summary		Upper level summary table of daily metrics grouped only by provider and sortendtime.	Based on data in segmented dist table. State='Shipped' and usertype=2.
14	rp_dtu_summary_req_old rp_dtu_summary_ord_old rp_dtu_summary_old		Work tables used to generate portion of rp_dtu_summary table that comes from offline dist segments .	
15	rp_dtu_summary_req_new rp_dtu_summary_ord_new rp_dtu_summary_new		Work tables used to generate portion of rp_dtu_summary table that comes from online dist segments	
16	rp_dfa_by_subscription		Summary table of daily metrics at the datatype level for all subscription data distributed from archives	Based on data in segmented dist table. State='Shipped'. Grouped by category, rollup and userspecified as well as other fields.
17	rp_dfa_by_subscription_old rp_dfa_by_subscription_ne w		Work tables used to generate portions of rp_dfa_by_subscription table that come from offline and online dist segments, resp.	
18	rp_dfa_instrument_last30days_(req,ord) rp_dfa_instrument_last30da		Work tables used to refresh last 30 days of rp_dfa_instrument_datatype table during weekday processing	

Table ID	EDGRS Table Name	Primary Key	EDGRS Table Description	Comments
	ys_work			
19	tempdist2_work		Work table used to refresh last 30 days of rp_dfa_by_subscription table during weekday processing	
20	rp_order_status		Table containing order status metrics for distributed data	Derived from segmented dist tale
21	rp_order_status_(old,new)		Work tables used to generate portions of rp_order_status table that come from offline and online dist segments, resp.	
22	orderspancnt_(old,new)		Tables identifying orders that span multiple days for offline and online dist segments, resp.	Offline table is generated when first created or after backfill. Online table is regenerated each time as part of daily processing.
23	orderspandetail_(old,new)		Tables containing detail dist records for orders that span multiple days	Offline table is generated when first created or after backfill. Online table is regenerated each time as part of daily processing. Tables are used in correcting order counts generated in dist ad-hoc reports (See Section 5.2.6)
24	orderspandetail_work		Work table used to generate orderspandetail_(old,new) tables.	
25	rp_dtu_orders		Table containing order counts for distributions to end users	
26	rp_dtu_orders_(old,new)		Work tables used to generate portions of rp_dtu_orders table that come from offline and online dist segments, resp	
27	graph_dist		Table containing daily dfa and dtu metrics	Grouped by provider, instrument, dtlevel and discipline
28	graph_dist_avg		Table containing daily dfa and dtu metrics	Grouped by provider, instrument and dtlevel

*Table 5-29 Index Definitions for Tables Used to Ingest Data into dist*

Index ID	EDGRS Index Name	Table	EDGRS Index Description	Clust-ered	Unique Index	Comments
1	INDX_insertime	rp_dfa_instrument_dsatatype	Indexed on insertime	Yes	Yes	For speed

<b>Index ID</b>	<b>EDGRS Index Name</b>	<b>Table</b>	<b>EDGRS Index Description</b>	<b>Clust-ered</b>	<b>Unique Index</b>	<b>Comments</b>
2	PK_dist_2 PK_dist_xx_1 PK_dist_04S1_0	dist EdgrsDist.dbo.distxx_1 EdgrsDist.dbo.dist04S1	Primary Key	Yes	Yes	
3	INDX_sortendtime INDX_sortendtime_xx_1 INDX_sortendtime_04S1_0	dist EdgrsDist.dbo.distxx EdgrsDist.dbo.dist04S1	indexed on sortendtime	No	No	For speed

#### 5.5.4 Building the EDGRS SCRS Tables

The SCRS tables provide user log metrics for various types of online accesses by users to ECS daacs.

Notes:

- All procedures and tables described below are in the EdgrsCurrent database except where indicated otherwise.

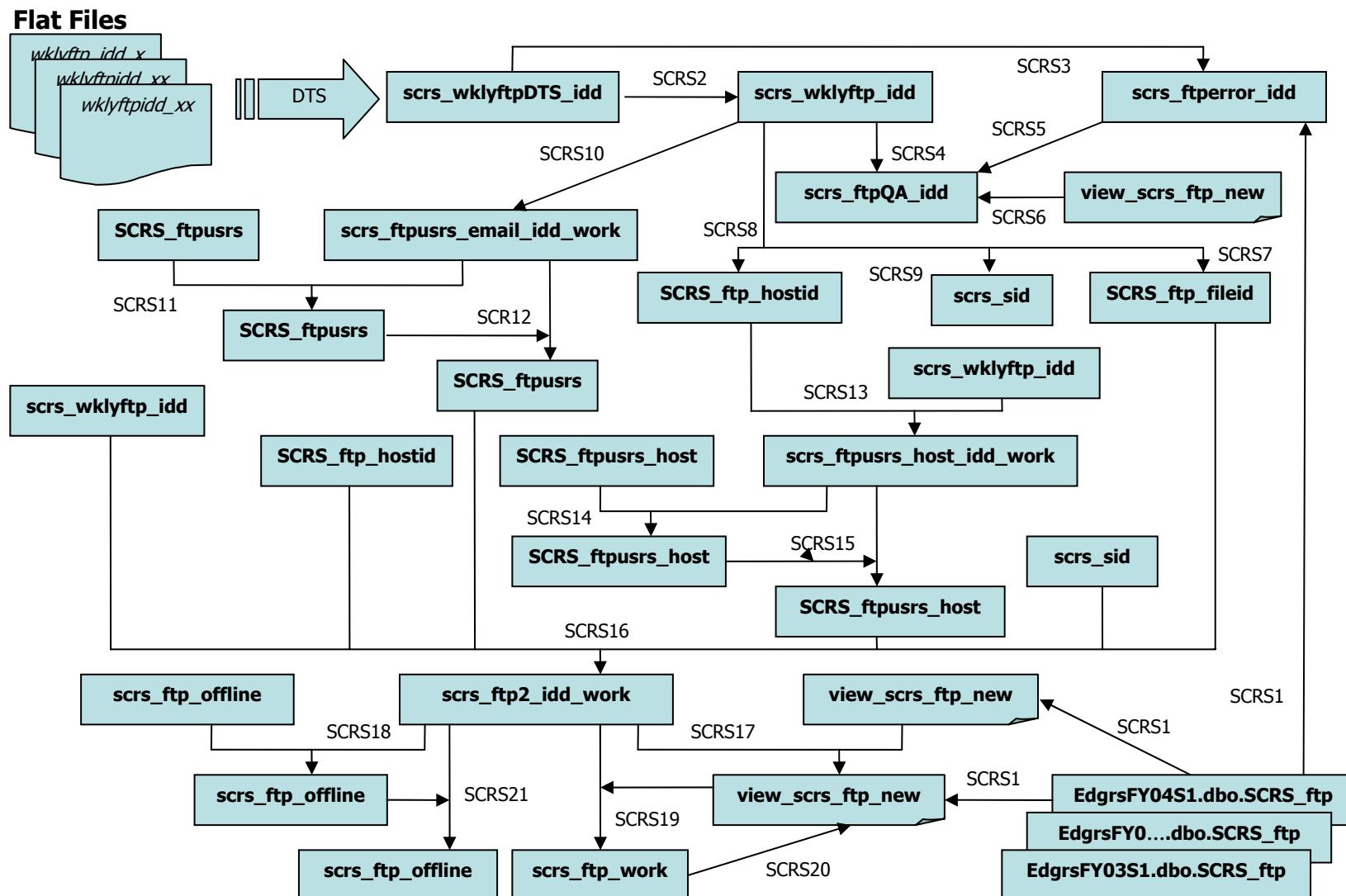


Figure 5-15 SCRS Ingest – ftp DFD

**Table 5-30 SCRS Ingest – ftp DFD Key**

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
SCRS1	view_scrs_ftp_new	create_scrs_views.sql	Group online segments via UNION ALL	View of online segments for SCRS_ftp table
SCRS2	pro_scrs_ingest_ftp_idd	Update_scrs_ftp_curr.sql	Reformat the input data into scrs_wklyftp_idd table	
SCRS3	pro_scrs_ingest_ftp_idd	Update_scrs_ftp_curr.sql	Check for erroneous records and add info to scrs_ft perror_idd table	
SCRS4	pro_scrs_ingest_ftp_idd	Update_scrs_ftp_curr.sql	Add summary QA metrics generated from scrs_wklyftp_idd to the scrs_ftpQA_idd table	
SCRS5	pro_scrs_ingest_ftp_idd	Update_scrs_ftp_curr.sql	Add summary error metrics generated from scrs_ft perror_idd to the scrs_ftpQA_idd table	
SCRS6	pro_scrs_ingest_ftp_idd	Update_scrs_ftp_curr.sql	Add record count and min and max date metrics from the online segments of SCRS_ftp to the scrs_ftpQA_idd table	.
SCRS7	pro_scrs_update_ftp_idd	Update_scrs_ftp_curr.sql	Add new filenames to the SCRS_ftp_fileid table	Done for normalization
SCRS8	pro_scrs_update_ftp_idd	Update_scrs_ftp_curr.sql	Add new hosts to the SCRS_ftp_host table	Done for normalization
SCRS9	pro_scrs_update_ftp_idd	Update_scrs_ftp_curr.sql	Add new sids to the scrs_sid table	Done for normalization
SCRS10	pro_scrs_update_ftp_idd	Update_scrs_ftp_curr.sql	Get email address info from input data and store in scrs_ftpusrs_email_idd_work	
SCRS11	pro_scrs_update_ftp_idd	Update_scrs_ftp_curr.sql	Update existing records in SCRS_ftpusrs table with new	If new firstdate is earlier

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
			Firstdate in scrs_ftpusrs_email_idd_work	
SCRS12	pro_scs_update_ftp_idd	Update_scs_ftp_curr.sql	Insert new records from scrs_ftpusrs_email_idd_work into SCRS_ftpusrs table	
SCRS13	pro_scs_update_ftp_idd	Update_scs_ftp_curr.sql	Get distinct host info from input data and store in scrs_ftpusrs_host_idd_work table	Joined with SCRS_ftp_hostid
SCRS14	pro_scs_update_ftp_idd	Update_scs_ftp_curr.sql	Update existing records in SCRS_ftpusrs_host with new Firstdate from scrs_ftpusrs_host_idd_work	If new firstdate is earlier
SCRS15	pro_scs_update_ftp_idd	Update_scs_ftp_curr.sql	Insert new records from scrs_ftpusrs_host_idd_work into SCRS_ftpusrs_host	
SCRS16	pro_scs_update_ftp_idd	Update_scs_ftp_curr.sql	Get distinct ftp data from scrs_wklyftp_idd and store in work table scrs_ftp2_idd_work	Joined with SCRS_ftp_hostid, scrs_sid, SCRS_ftpusrs, SCRS_ftpusrs_host
SCRS17	pro_scs_update_ftp_idd	Update_scs_ftp_curr.sql	Update existing records in the online segments of the SCRS_ftp table with data from scrs_ftp2_idd_work	Joined with online segments in view_scs_ftp_new
SCRS18	pro_scs_update_ftp_idd	Update_scs_ftp_curr.sql	Update existing records in scrs_ftp_offline with data from scrs_ftp2_idd_work	scrs_ftp_offline contains data earlier than data in view_scs_ftp_new
SCRS19	pro_scs_update_ftp_idd	Update_scs_ftp_curr.sql	Insert new records from scrs_ftp2_idd_work into scrs_ftp_work	
SCRS20	pro_scs_update_ftp_idd	Update_scs_ftp_curr.sql	Add new records from scrs_ftp_work into view_scs_ftp_new (online segments)	

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
SCRS21	pro_scs_update_ftp_idd	Update_scs_ftp_curr.sql	Add new records for offline data to scrs_ftp_offline	
SCRSxx	pro_scs_update_ftp_idd	Update_scs_ftp_curr.sql	Add record count, min and max date to scrs_ftpQA_idd	

*Table 5-31 Definitions for Tables Used to Build the SCRS\_ftp Table*

Table ID	EDGRS Table Name	Primary Key/Identity	EDGRS Table Description	Comments
1	scrs_wklyftpDTS_idd		staging area for SCRS_ftp data transferred from flat files (log input)	The structure of this file mimics the structure of the input data.
2	scrs_ft perror_idd		Intermediate table containing errors found in input data.	
3	scrs_ftpQA_idd		Intermediaite table containing QA information about input data	
4	scrs_ftpusrs_email_idd_work scrs_ftpusrs_host_idd_work scrs_wklyftp_idd scrs_ftp2_idd_work scrs_ftp_work		Work tables	.
5	SCRS_ftp_hostid	hostname	Table containing names of servers involved in ftp transfers.	Used for normalization
6	SCRS_ftp_fileid	filename/fileid	Table containing pathnames of files transferred	Used for normalization
7	scrs_sid	source_name source_txt/	Table containing information about data sources	Used for normalization

Table ID	EDGRS Table Name	Primary Key/ Identity	EDGRS Table Description	Comments
		sid		
8	SCRS_ftpusrs	addr	Table containing addresses of users doing ftp	Used for normalization
9	SCRS_ftpusrs_host	addr hostid	Table containing additional information about users doing ftp	Used for normalization
10	scrs_ftp_offline		Intermediate table holding ftp records received that precede the offline date threshold	Currently that date is 10-01-2002
11	SCRS_ftp EdgrsFY0x.dbo.SCRS_ftp	daac, sid, ftp_dt, ftp_tm, fileid, hostid, cust_id, cust_host, datatype	Base segmented table holding data derived from ftp logs	

**Table 5-32 Index Definitions for Tables Used to Build the SCRS\_ftp Table**

Index ID	EDGRS Index Name	Table	EDGRS Index Description	Clust- ered	Unique Index	Comments
1	PK_SCRS_ftp00 PK_SCRS_ftp	SCRS_ftp EdgrsFY0x.dbo.SCRS_ftp	Primary Key	Yes	Yes	
2	INDX_SCRS_ftp_date00 INDX_SCRS_ftp_date	SCRS_ftp EdgrsFY0x.dbo.SCRS_ftp	Index on ftp_dt	No	No	
3	PK_SCRS_sid	scrs_sid	Primary Key	Yes	Yes	
4	PK_SCRS_ftp_hostid	SCRS_ftp_hostid	Primary Key	No	Yes	
5	PK_SCRS_ftpusrs	SCRS_ftpusrs	Primary Key	No	Yes	
6	CINDX_SCRS_ftpusrs	SCRS_ftpusrs	Index on cust_id	Yes	Yes	

Index ID	EDGRS Index Name	Table	EDGRS Index Description	Clustered	Unique Index	Comments
7	PK_SCRS_ftp_fileid	SCRS_ftp_fileid	Primary Key	No	Yes	
8	PK_SCRS_ftpusrs_host	SCRS_ftpusrs_host	Primary Key	Yes	Yes	

### Flat Files

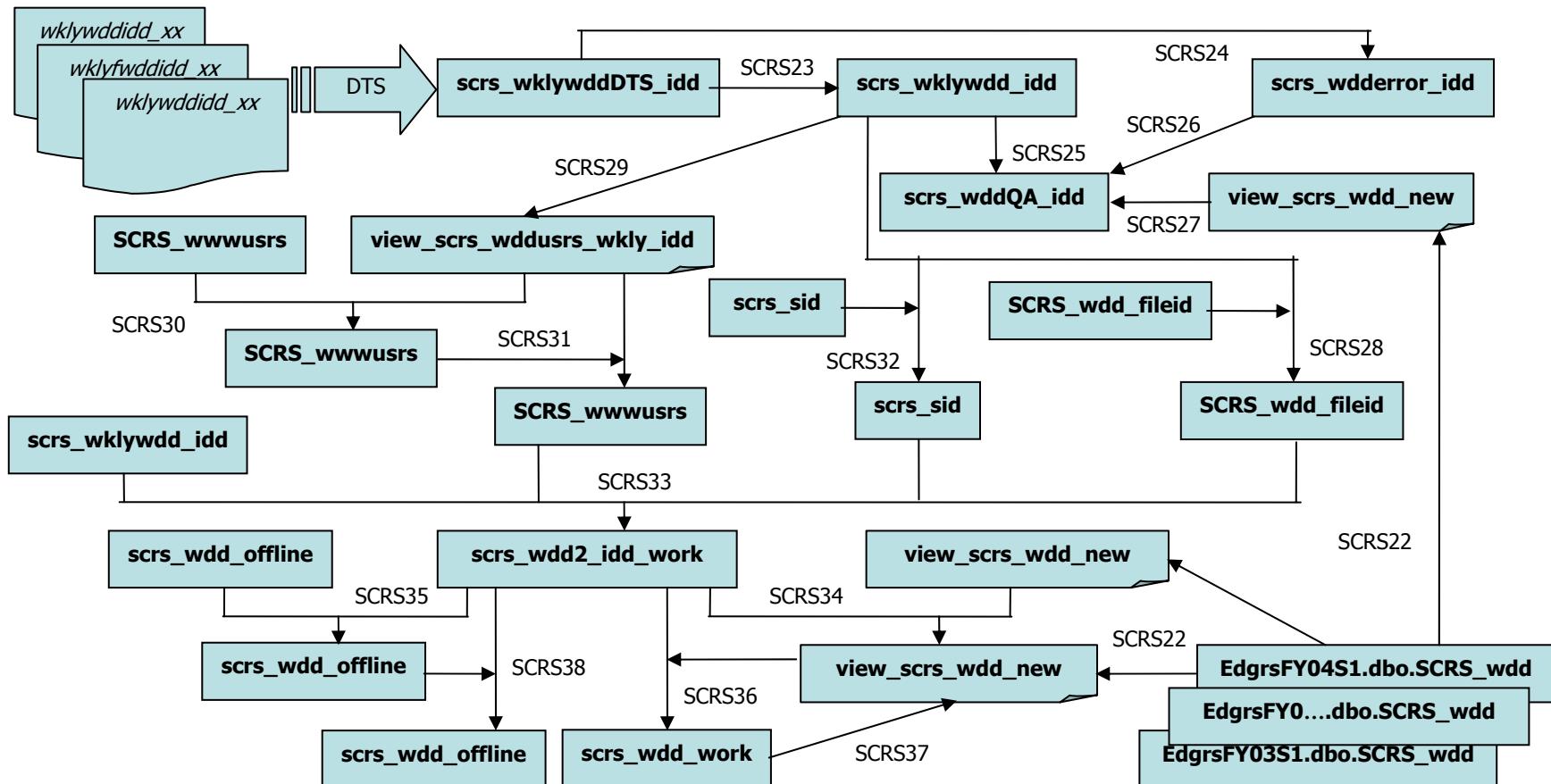


Figure 5-16 SCRS Ingest – wdd DFD

**Table 5-33 SCRS Ingest – wdd DFD Key**

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
SCRS22	view_scrs_wdd_new	create_scrs_views.sql	Group online segments via UNION ALL	View of online segments for SCRS_wdd table
SCRS23	pro_scrs_ingest_wdd_idd	Update_scrs_wdd_curr.sql	Reformat the input data into scrs_wklywdd_idd table	
SCRS24	pro_scrs_ingest_wdd_idd	Update_scrs_wdd_curr.sql	Check for erroneous records and add info to scrs_wdderror_idd table	
SCRS25	pro_scrs_ingest_wdd_idd	Update_scrs_wdd_curr.sql	Add summary QA metrics generated from scrs_wklywdd_idd to the scrs_wddQA_idd table	
SCRS26	pro_scrs_ingest_wdd_idd	Update_scrs_wdd_curr.sql	Add summary error metrics generated from scrs_wdderror_idd to the scrs_wddQA_idd table	
SCRS27	pro_scrs_ingest_wdd_idd	Update_scrs_wdd_curr.sql	Add record count, min and max date metrics from the online segments (before this run) of the SCRS_wdd table to the scrs_wddQA_idd table	.
SCRS28	pro_scrs_update_wdd_idd	Update_scrs_wdd_curr.sql	Add new filenames to the SCRS_wdd_fileid table	Done for normalization
SCRS29	view_scrs_wddusrs_wkly_idd	Update_scrs_wdd_curr.sql	Get host info from input data	Gets info from scrs_wklywdd_idd
SCRS30	pro_scrs_update_wdd_idd	Update_scrs_wdd_curr.sql	Update existing records in SCRS_wwwusrs table with new Firstdate info provided by view_scrs_wddusrs_wkly_idd	If new firstdate is earlier
SCRS31	pro_scrs_update_wdd_idd	Update_scrs_wdd_curr.sql	Insert new records from view_scrs_wddusrs_wkly_idd into SCRS_wwwusrs table	
SCRS32	pro_scrs_update_wdd_idd	Update_scrs_wdd_curr.sql	Add new sids to the scrs_sid table	Done for normalization
SCRS33	pro_scrs_update_wdd_idd	Update_scrs_wdd_curr.sql	Get wdd metrics data from scrs_wklywdd_idd and store in work table scrs_wdd2_idd_work	Joined with SCRS_wdd_fileid, scrs_sid, SCRS_wwwusrs

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
SCRS34	pro_scrs_update_wdd_idd	Update_scrs_wdd_curr.sql	Update existing records in the online segments of the SCRS_wdd table with data from scrs_wdd2_idd_work	Joined with online segments in view_scrs_wdd_new
SCRS35	pro_scrs_update_wdd_idd	Update_scrs_wdd_curr.sql	Update existing records in scrs_wdd_offline with data from scrs_wdd2_idd_work	
SCRS36	pro_scrs_update_wdd_idd	Update_scrs_wdd_curr.sql	Insert new records from scrs_wdd2_idd_work into scrs_wdd_work	
SCRS37	pro_scrs_update_wdd_idd	Update_scrs_wdd_curr.sql	Add new records from scrs_wdd_work into view_scrs_wdd_new	
SCRS38	pro_scrs_update_wdd_idd	Update_scrs_wdd_curr.sql	Add new records for offline data to scrs_wdd_offline	
SCRSxx	pro_scrs_update_wdd_idd	Update_scrs_wdd_curr.sql	Add record count, min and max date to scrs_wddQA_idd	

**Table 5-34 Definitions for Tables Used to Build the SCRS\_wdd Table**

Table ID	EDGRS Table Name	Primary Key/ Identity	EDGRS Table Description	Comments
1	scrs_wklywddDTS_idd		staging area for SCRS_wdd data transferred from flat files (log input) .	The structure of this file mimics the structure of the input data.
2	scrs_wdderror_idd		Intermediate table containing errors found in input data.	
3	scrs_wddQA_idd		Intermediaite table containing QA information about input data	
4	scrs_wklywdd_idd scrs_wdd2_idd_work scrs_wdd_work		Work tables .	.
5	SCRS_wdd_fileid	filename/ fileid	Table containing pathnames of files transferred	Used for normalization

6	scrs_sid	source_name source_txt/ sid	Table containing information about data sources	Used for normalization
7	SCRS_wwwusrs	host/ cust_id	Table containing addresses of users doing wdd transfers	Used for normalization
8	scrs_wdd_offline		Intermediate table holding wdd records received that precede the offline date threshold	Currently that date is 10-01-2002
9	SCRS_wdd EdgrsFY0x.dbo.SCRS_wdd	daac, sid, www_dt, www_tm, fileid, cust_id, datatype	Base segmented table holding data derived from wdd logs	

*Table 5-35 Index Definitions for Tables Used to Build the SCRS\_wdd Table*

Index ID	EDGRS Index Name	Table	EDGRS Index Description	Clust-ered	Unique Index	Comments
1	PK_SCRS_wdd00 PK_SCRS_wdd	SCRS_wdd EdgrsFY0x.dbo.SCRS_wdd	Primary Key	Yes	Yes	
2	INDX_SCRS_wdd_date00 INDX_SCRS_wdd_date	SCRS_wdd EdgrsFY0x.dbo.SCRS_wdd	Index on www_dt	No	No	
3	PK_SCRS_sid	scrs_sid	Primary Key	Yes	Yes	
4	PK_SCRS_wwwusrs	SCRS_wwwusrs	Primary Key	No	Yes	
5	CINDX_SCRS_wwwusrs	SCRS_wwwusrs	Index on cust_id	Yes	Yes	
6	PK_SCRS_wdd_fileid	SCRS_wdd_fileid	Primary Key	No	Yes	

## Flat Files

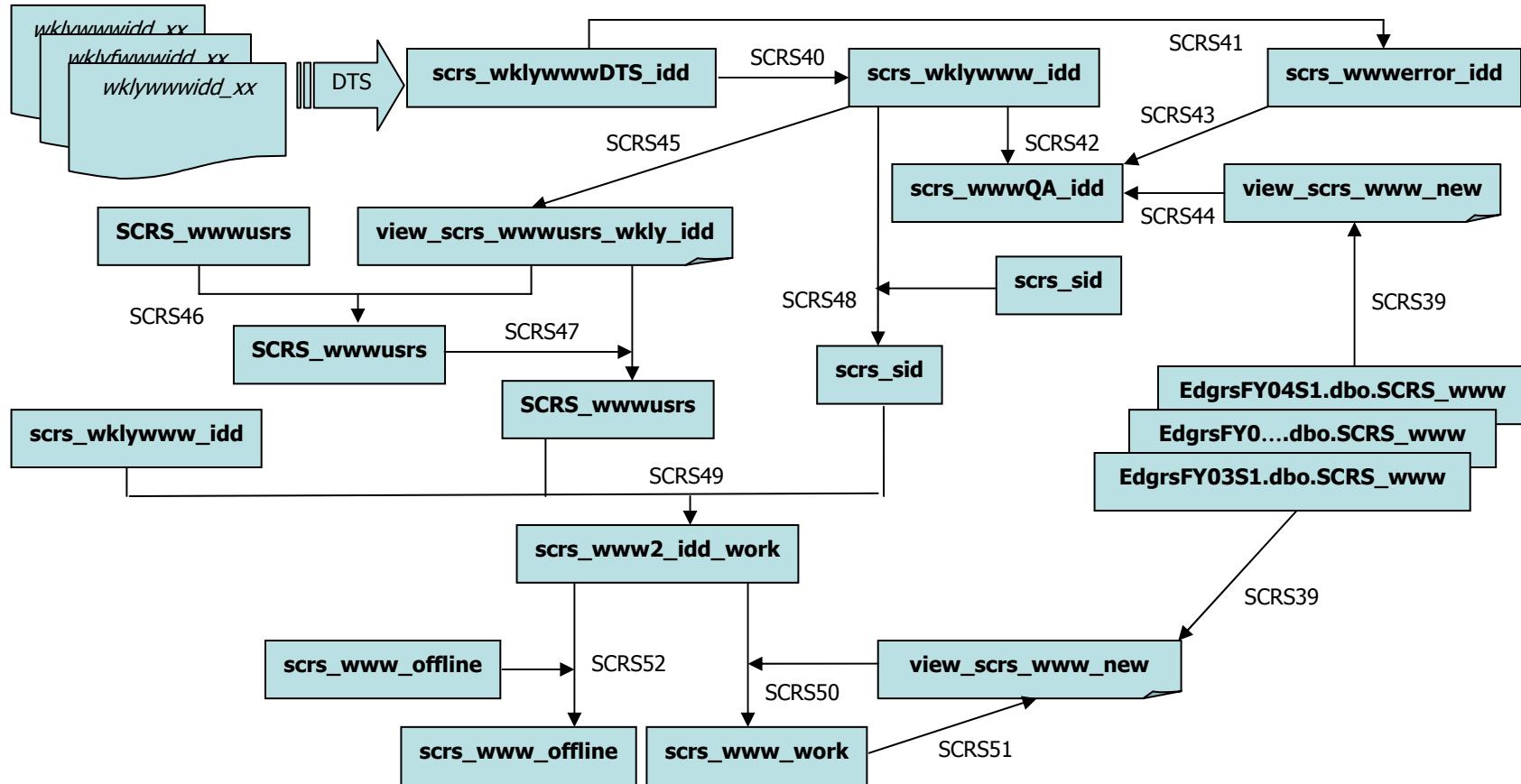


Figure 5-17 SCRS Ingest – www DFD

**Table 5-36 SCRS Ingest – www DFD Key**

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
SCRS39	view_scrs_www_new	create_scrs_views.sql	Group online segments via UNION ALL	View of online segments for SCRS_www table
SCRS40	pro_scrs_ingest_www_idd	Update_scrs_www_curr.sql	Reformat the input data to generate the scrs_wklywww_idd table	
SCRS41	pro_scrs_ingest_www_idd	Update_scrs_www_curr.sql	Check for erroneous records and add info to scrs_wwwerror_idd table	
SCRS42	pro_scrs_ingest_www_idd	Update_scrs_www_curr.sql	Add summary QA metrics generated from scrs_wklywww_idd to the scrs_wwwQA_idd table	
SCRS43	pro_scrs_ingest_www_idd	Update_scrs_www_curr.sql	Add summary error metrics generated from scrs_wwwerror_idd to the scrs_wwwQA_idd table	
SCRS44	pro_scrs_ingest_www_idd	Update_scrs_www_curr.sql	Add record count, min and max date metrics from the online segments (BEFORE this run) of the SCRS_www table to the scrs_wwwQA_idd table	
SCRS45	view_scrs_wwwusrs_wkly_idd	Update_scrs_www_curr.sql	Get distinct host and earliest date info from scrs_wklywww_idd	
SCRS46	pro_scrs_update_www_idd	Update_scrs_www_curr.sql	Update existing records in SCRS_wwwusrs table with firstdate provided by view_scrs_wwwusrs_wkly_idd	Where www_dt>Firstdate
SCRS47	pro_scrs_update_www_idd	Update_scrs_www_curr.sql	Insert new records from view_scrs_wwwusrs_wkly_idd into SCRS_wwwusrs table	

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
SCRS48	pro_scs_update_www_idd	Update_scs_www_curr.sql	Add new sids to the scrs_sid table	Done for normalization
SCRS49	pro_scs_update_www_idd	Update_scs_www_curr.sql	Get www distinct normalized records from scrs_wklywww_idd and store in work table scrs_www2_idd_work	Joined with scrs_sid
SCRS50*	pro_scs_update_www_idd	Update_scs_www_curr.sql	Insert new records from scrs_www2_idd_work into scrs_www_work	
SCRS51*	pro_scs_update_www_idd	Update_scs_www_curr.sql	Add new records from scrs_www_work into SCRS_www	Where www_dt>'1993-01-01' (i.e. valid)
SCRS52*	pro_scs_update_www_idd	Update_scs_www_curr.sql	Add new records for offline data to scrs_www_offline	
SCRSxx	pro_scs_update_www_idd	Update_scs_www_curr.sql	Add record count, min and max date AFTER this run into scrs_wwwQA_idd	For QA purposes

\* Update of existing records for online and offline data is missing

**Table 5-37 Definitions for Tables Used to Build the SCRS\_www Table**

Table ID	EDGRS Table Name	Primary Key/Identity	EDGRS Table Description	Comments
1	scrs_wklywwwDTS_idd		staging area for SCRS_www data transferred from flat files (log input)	The structure of this file mimics the structure of the input data.

Table ID	EDGRS Table Name	Primary Key/ Identity	EDGRS Table Description	Comments
2	scrs_wwwerror_idd		Intermediate table containing errors found in input data.	
3	scrs_wwwQA_idd		Intermediaite table containing QA information about input data	
4	scrs_wklywww_idd scrs_www2_idd_work scrs_www_work		Work tables	.
5	scrs_sid	source_name source_txt/ sid	Table containing information about data sources	Used for normalization
6	SCRS_wwwusrs	host/ cust_id	Table containing addresses of users doing www transfers	Used for normalization
7	scrs_www_offline		Intermediate table holding www records received that precede the offline date threshold	Currently that date is 10-01-2002
8	SCRS_www EdgrsFY0x.dbo.SCRS_www	daac, sid, www_dt, www_tm, fileid, cust_id, datatype	Base segmented table holding data derived from www logs	

*Table 5-38 Index Definitions for Tables Used to Build the SCRS\_www Table*

Index ID	EDGRS Index Name	Table	EDGRS Index Description	Clust-ered	Unique Index	Comments
1	PK_SCRS_www00 PK_SCRS_www	SCRS_www EdgrsFY0x.dbo.SCRS_www	Primary Key	Yes	Yes	
2	INDX_SCRS_www_date00 INDX_SCRS_www_date	SCRS_www EdgrsFY0x.dbo.SCRS_www	Index on www_dt	No	No	

<b>Index ID</b>	<b>EDGRS Index Name</b>	<b>Table</b>	<b>EDGRS Index Description</b>	<b>Clust-ered</b>	<b>Unique Index</b>	<b>Comments</b>
3	PK_SCRS_sid	scrs_sid	Primary Key	Yes	Yes	
4	PK_SCRS_wwwusrs	SCRS_wwwusrs	Primary Key	No	Yes	
5	CINDEX_SCRS_wwwusrs	SCRS_wwwusrs	Index on cust_id	Yes	Yes	

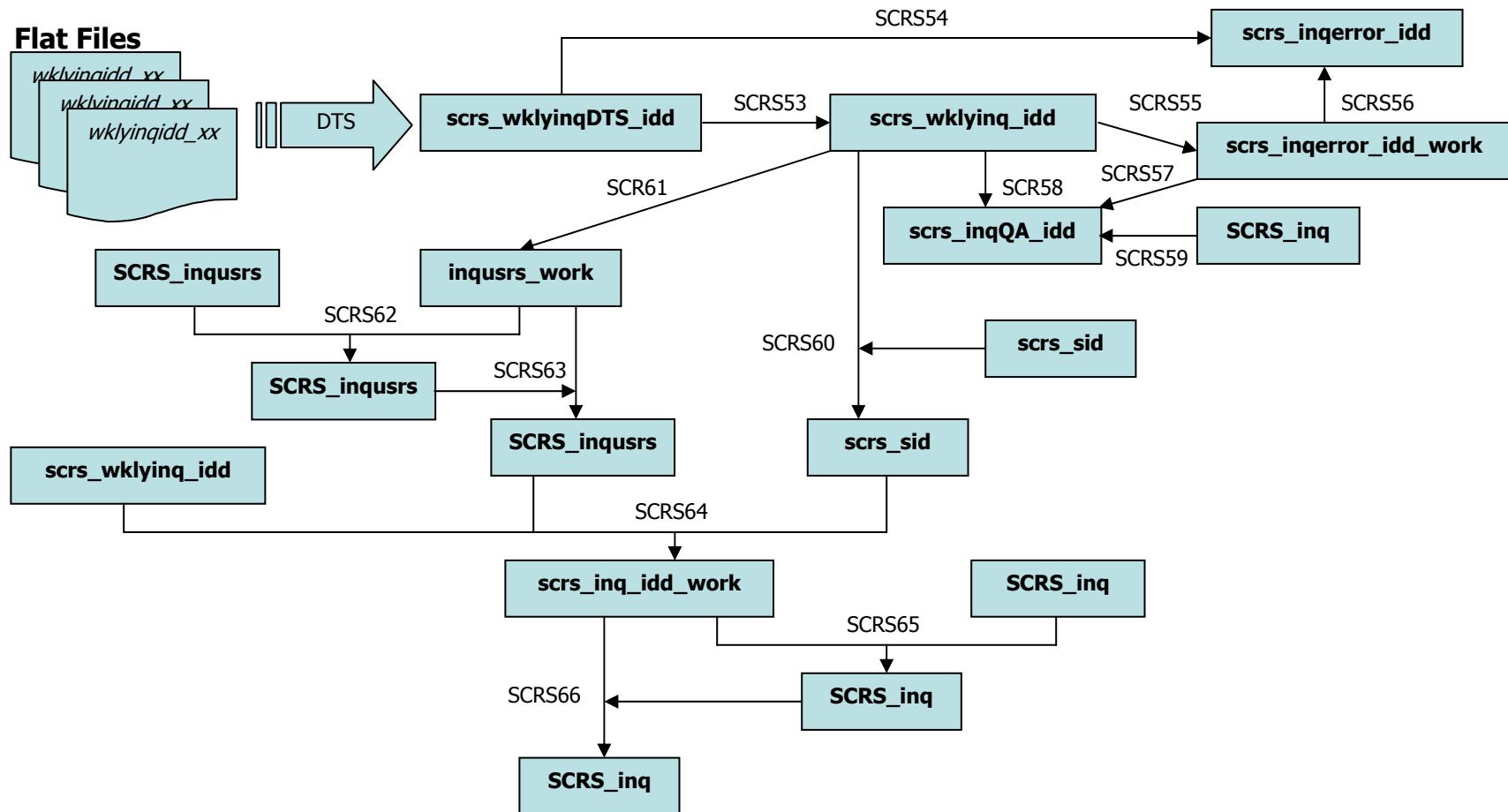


Figure 5-18 SCRS Ingest – inq DFD

**Table 5-39 SCRS Ingest – inq DFD Key**

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
SCRS53	pro_scs_ ingest_inq_idd	Update_scs_ inq_curr.sql	Reformat the input data into scrs_wklyinq_idd table	
SCRS54	pro_scs_ ingest_inq_idd	Update_scs_ inq_curr.sql	Check for erroneous records and add info to scrs_inqerror_idd_work table	
SCRS55	pro_scs_ ingest_www_idd	Update_scs_ inq_curr.sql	Get duplicate records from input data into scrs_inqerror_idd_work	
SCRS56	pro_scs_ ingest_inq_idd	Update_scs_ inq_curr.sql	Add those records to the scrs_inqerror_idd table	
SCRS57	pro_scs_ ingest_inq_idd	Update_scs_ inq_curr.sql	Add summary error metrics for those records to the scrs_inqQA_idd table	
SCRS58	pro_scs_ ingest_inq_idd	Update_scs_ inq_curr.sql	Add summary of min and max date metrics generated from scrs_wklyinq_idd to the scrs_inqQA_idd table	.
SCRS59	pro_scs_ ingest_inq_idd	Update_scs_ inq_curr.sql	Add record count, min and max date metrics BEFORE this run from the SCRS_inq table to the scrs_inqQA_idd table	
SCRS60	pro_scs_update_inq_idd	Update_scs_ inq_curr.sql	Add new sids to the scrs_sid table	Done for normalization
SCRS61	pro_scs_update_inq_idd	Update_scs_ inq_curr.sql	Get user info (min(inq_dt) for each lname/addr) into inqusrss_work from scrs_wklyinq_idd table	
SCRS62	pro_scs_update_inq_idd	Update_scs_ inq_curr.sql	Update existing records with new Firstdate in SCRS_inqusrss table with info from inqusrss_work	where inq_dt is earlier than Fristdate
SCRS63	pro_scs_update_inq_idd	Update_scs_ inq_curr.sql	Insert new records from inqusrss_work into SCRS_inqusrss table	

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
SCRS64	pro_scs_update_inq_idd	Update_scsr_inq_curr.sql	Get distinct normalized inq records from scrs_wklyinq_idd and store in work table scrs_inq_idd_work	Joined with scrs_sid and SCRS_inqusr
SCRS65	pro_scs_update_inq_idd	Update_scsr_inq_curr.sql	Update existing records in SCRS_inq table with data from scrs_inq_idd_work table	
SCRS66	pro_scs_update_inq_idd	Update_scsr_inq_curr.sql	Insert new records from scrs_inq_idd_work into SCRS_inq	
SCRSxx	pro_scs_update_inq_idd	Update_scsr_inq_curr.sql	Add record count, min and max date metrics AFTER this run from SCRS_inq table into scrs_inqQA_idd	

*Table 5-40 Definitions for Tables Used to Build the SCRS\_inq Table*

Table ID	EDGRS Table Name	Primary Key/Identity	EDGRS Table Description	Comments
1	scrs_wklyinqDTS_idd		staging area for SCRS_inq data transferred from flat files (log input)	The structure of this file mimics the structure of the input data.
2	scrs_inqerror_idd		Intermediate table containing errors found in input data.	
3	scrs_inqQA_idd		Intermediate table containing QA information about input data	
4	scrs_wklyinq_idd scrs_inq_idd_work		Work tables .	
5	scrs_sid	source_name source_txt/ sid	Table containing information about data sources	Used for normalization
6	SCRS_inqusr	lname	Table containing addresses of users doing inq accesses	Used for normalization

Table ID	EDGRS Table Name	Primary Key/ Identity	EDGRS Table Description	Comments
		addr		
7	SCRS_inq	daac, sid, inq_dt, inq_tm, cust_id	Base table holding data derived from inq logs	

***Table 5-41 Index Definitions for Tables Used to Build the SCRS\_inq Table***

Index ID	EDGRS Index Name	Table	EDGRS Index Description	Clust-ered	Unique Index	Comments
1	PK_SCRS_inq	SCRS_inq	Primary Key	Yes	Yes	
2	CINDX_SCRS_inq	SCRS_inq	Index on inq_dt	No	No	
3	PK_SCRS_sid	scrs_sid	Primary Key	Yes	Yes	
4	PK_SCRS_inqusr	SCRS_inqusr	Primary Key	No	Yes	

### 5.5.5 Building the EDGRS SCRS SummaryTables

The base SCRS tables, SCRS\_ftp, SCRS\_wdd, SCRS\_www and SCRS\_inq, contain user data extracted from access logs. This data is used in generating periodic user characterization reports.

The summary tables are built from the SCRS base tables by grouping the data in the tables over the various fields including the date each access occurred. Use of these tables speeds up generation of the reports.

Notes:

- All procedures and tables described below are in the EdgrsCurrent database except where indicated otherwise.

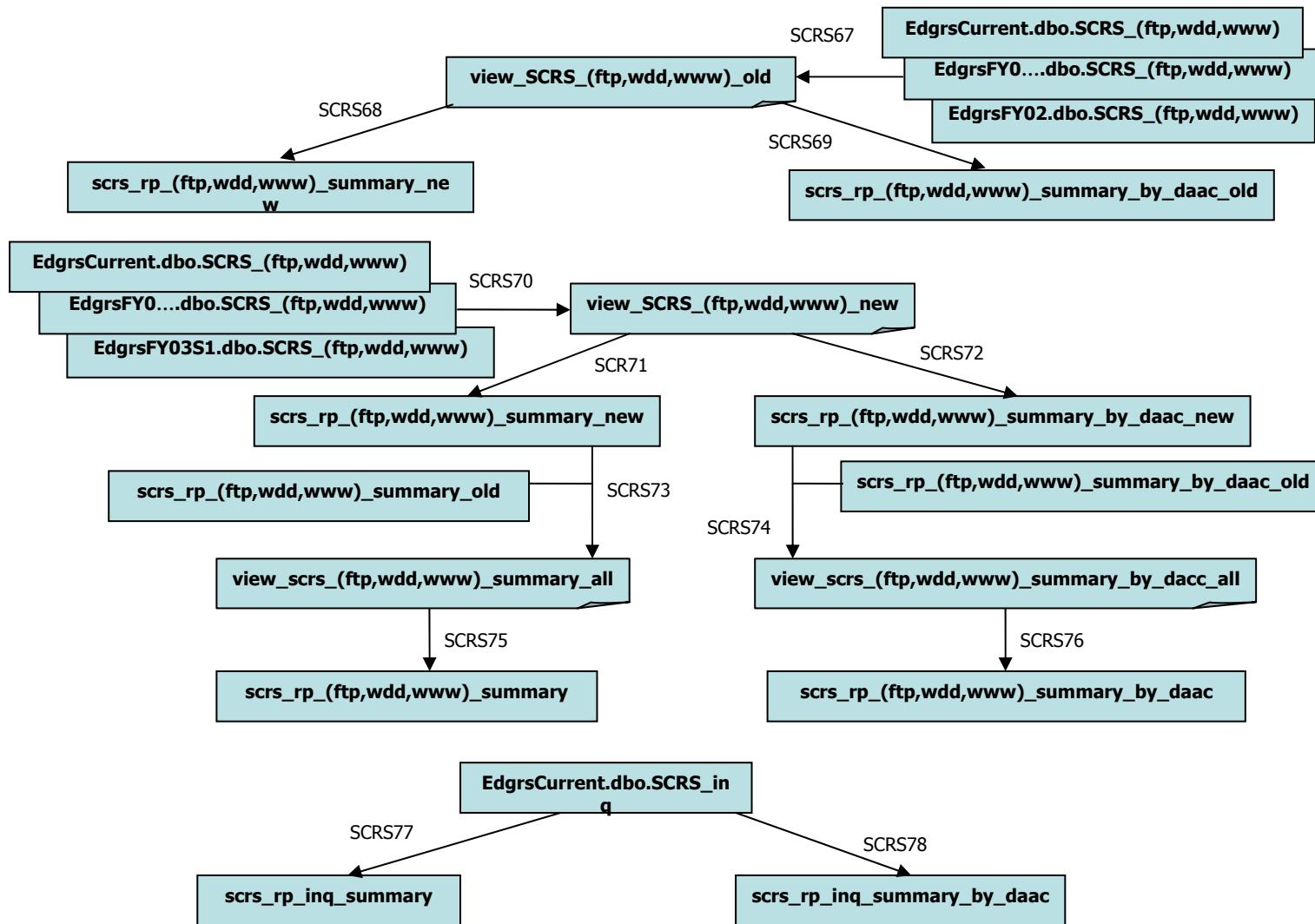


Figure 5-19 SCRS Report Generation DFD

**Table 5-42 SCRS Report Generation DFD Key**

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
SCRS67*	view_SCRS_ftp_old view_SCRS_wdd_old view_SCRS_www_old	Create_scrs_views.sql	View of online segments of SCRS_ftp View of online segments of SCRS_wdd View of online segments of SCRS_www	Generated from offline segments via UNION ALL
SCRS68*	pro_scrs_rp_ftp_summary_old pro_scrs_rp_wdd_summary_old pro_scrs_rp_www_summary_old	SCRS_ftp_curr_old.sql SCRS_wdd_curr_old.sql SCRS_www_curr_old.sql	Build scrs_rp_xxx_summary_old tables by grouping over daac, <date> and other fields (if any)	
SCRS69*	pro_scrs_rp_ftp_summary_old pro_scrs_rp_wdd_summary_old pro_scrs_rp_www_summary_old	SCRS_ftp_curr_old.sql SCRS_wdd_curr_old.sql SCRS_www_curr_old.sql	Build scrs_rp_xxx_summary_by_daac_old table by grouping over daac and <date>	
SCRS70	view_SCRS_ftp_new view_SCRS_wdd_new view_SCRS_www_new	Create_scrs_views.sql	View of online segments of SCRS_ftp View of online segments of SCRS_wdd View of online segments of SCRS_www	Generated from online segments via UNION ALL
SCRS71	pro_scrs_rp_ftp_summary_new pro_scrs_rp_wdd_summary_new pro_scrs_rp_www_summary_new	SCRS_ftp_curr_new.sql SCRS_wdd_curr_new.sql SCRS_www_curr_new.sql	Build scrs_rp_xxx_summary_new tables by grouping over daac, <date> and other fields (if any)	
SCRS72	pro_scrs_rp_ftp_summary_new pro_scrs_rp_wdd_summary_new pro_scrs_rp_www_summary_new	SCRS_ftp_curr_new.sql SCRS_wdd_curr_new.sql SCRS_www_curr_new.sql	Build scrs_rp_xxx_summary_by_daac_new table by grouping over daac and <date>	

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
SCRS73	view_scs_ftp_summary_all view_scs_wdd_summary_all view_scs_www_summary_all	Create_scs_views.sql	Merges new and old parts of scrs_rp_xxx_summary tables	Generated via UNION ALL
SCRS74	view_scs_ftp_summary_by_daac_all view_scs_wdd_summary_by_daac_all view_scs_www_summary_by_daac_all	SCRS_ftp_curr_all.sql SCRS_wdd_curr_all.sql SCRS_www_curr_all.sql	Merges new and old parts of scrs_rp_xxx_summary_by_daac tables	Generated via UNION ALL
SCRS75	pro_scs_rp_ftp_summary_all pro_scs_rp_wdd_summary_all pro_scs_rp_www_summary_all	SCRS_ftp_curr_all.sql SCRS_wdd_curr_all.sql SCRS_www_curr_all.sql	Build scrs_rp_xxx_summary tables by merging old and new parts	Used to display daacs horizontally in reports.
SCRS76	pro_scs_rp_ftp_summary_by_daac_all pro_scs_rp_wdd_summary_by_daac_all pro_scs_rp_www_summary_by_daac_all	SCRS_ftp_curr_all.sql SCRS_wdd_curr_all.sql SCRS_www_curr_all.sql	Build scrs_rp_xxx_summary_by_daac tables by merging old and new parts	Used to display daacs vertically in reports.
SCRS77	pro_scs_rp_inq_summary	SCRS_inq_curr.sql	Build scrs_rp_inq_summary table by grouping over daac and <date>	Used to display daacs horizontally in reports.
SCRS78	pro_scs_rp_inq_summary_by_daac	SCRS_inq_curr.sql	Build scrs_rp_inq_summary table by grouping over daac and <date>	Used to display daacs vertically in reports.

\* These procedures are not part of daily processing. They were run once to generate the old parts of the SCRS summary tables and cannot be run now because the segments they require are no longer accessible.

**Table 5-43 Definitions for Tables Used to Build SCRS Summary Tables**

Table ID	EDGRS Table Name	Primary Key	EDGRS Table Description	Comments
1	scrs_rp_xxx_summary_,old		Old part of scrs_rp_xxx_summary table representing data generated from offline segments	This part is static and currently not changeable.
2	scrs_rp_xxx_summary_,new		New part of scrs_rp_xxx_summary table representing data generated from online	This part is dynamic and is regenerated during daily processing.

Table ID	EDGRS Table Name	Primary Key	EDGRS Table Description	Comments
			segments	
3	scrs_rp_xxx_summary_,by_daac_old		Old part of scrs_rp_xxx_summary_by_daac table representing data generated from offline segments	This part is static and currently not changeable.
4	scrs_rp_xxx_summary_by_daac_new		New part of scrs_rp_xxx_summary_by_daac table representing data generated from online segments	This part is dynamic and is regenerated during daily processing.
5	scrs_rp_xxx_summary		Summary table holding data grouped by date with daacs vertical	
6	scrs_rp_xxx_summary_,by_daac		Summary table holding data grouped by date with daacs horizontal	

**Table 5-44 Index Definitions for Tables Used to Build SCRS Summary Tables**

Index ID	EDGRS Index Name	Table	EDGRS Index Description	Clust-ered	Unique Index	Comments

### 5.5.6 Building the EDGRS User Characterization Tables

The user characterization tables usrprofile\_curr and users are used to define users via lastname and emailaddr based on edrsid. edrsid is a unique identity field generated in the users table. Its value is passed on to tables in the dist subsystem either directly via lastname and emailaddr or via provider and userid through the usrprofile\_curr table.

Notes:

- All procedures and tables described below are in the EdgrsCurrent database except where indicated otherwise.

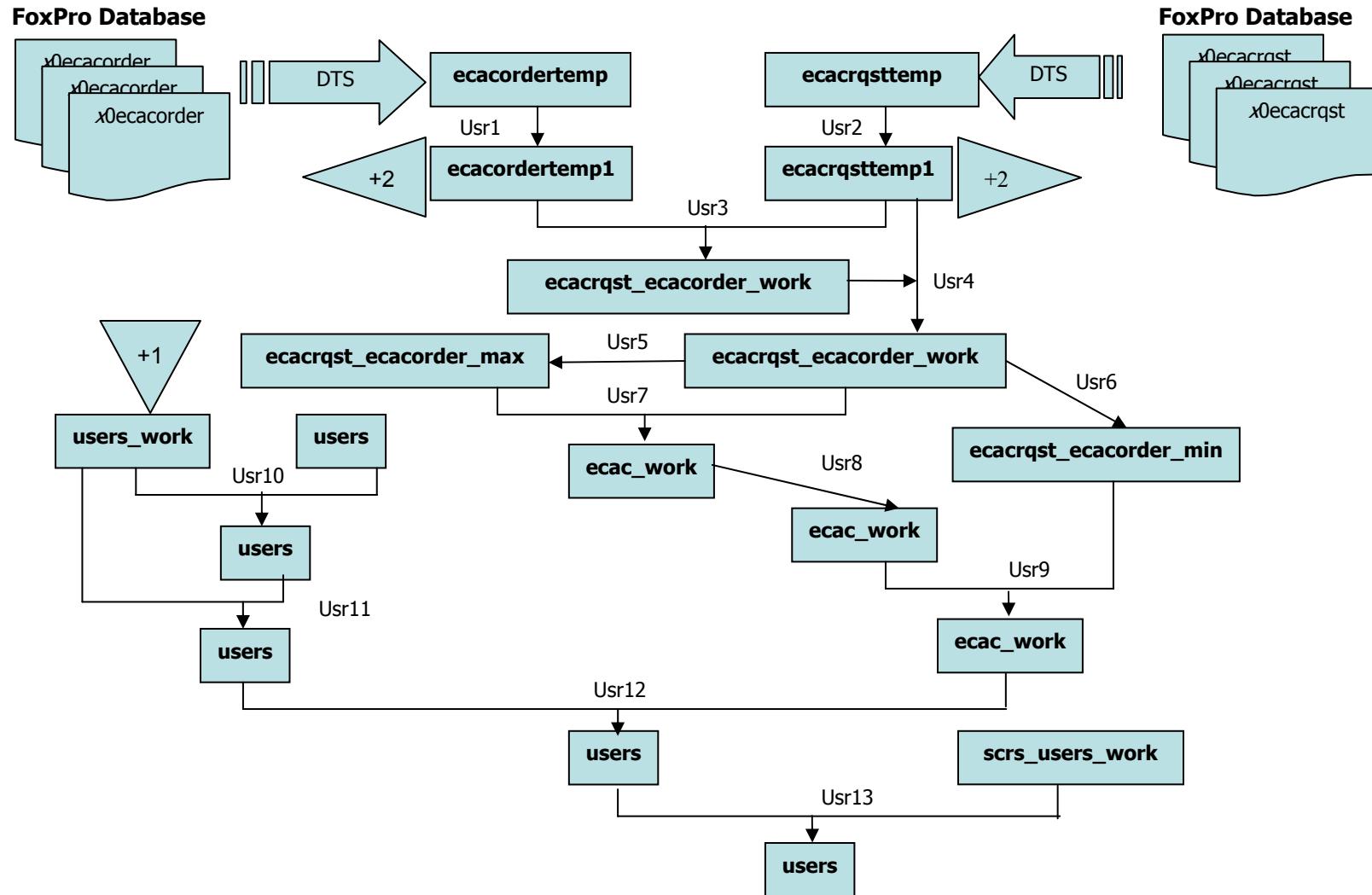


Figure 5-20 ecac Users – Ingest DFD

**Table 5-45 ecac Users – Ingest DFD Key**

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
Usr1	pro_update_ecac	Update_ecac.sql	Copy all records	Capitalize firstname, lastname and convert emailaddr to lowercase. Set lastname and emailaddr to null string ("") if null or 'NULL'
Usr2	pro_update_ecac	Update_ecac.sql	Copy all records	Capitalize firstname, lastname and convert emailaddr to lowercase. Set lastname and emailaddr to null string ("") if null or 'NULL'
Usr3	pro_update_ecac	Update_ecac.sql	Merge order and request information	Left outer join on ecacordtemp1. Derive calcaffili from emailaddr and firstdate from receivetim
Usr4	pro_update_ecac	Update_ecac.sql	Insert records with lastname+emailaddr not in ecacordertemp1	Default userid to ". Derive calcaffili from emailaddr and firstdate from receivetim
Usr5	pro_update_ecac	Update_ecac.sql	Generate ecacrqst_ecacorder_max auxiliary table to extract latest unique records based on lastname and emailaddr	distinct lastname, emailaddr, max(putnedgrs) from ecacrqst_ecacorder_work group by lastname,emailaddr
Usr6	pro_update_ecac	Update_ecac.sql	Generate ecacrqst_ecacorder_min auxiliary table to extract earliest unique records based on lastname and emailaddr	distinct lastname, emailaddr, min(receivetim) from ecacrqst_ecacorder_work group by lastname,emailaddr.
Usr7	pro_update_ecac	Update_ecac.sql	Apply ecacrqst_ecacorder_max to extract records based on lastname + emailaddr with latest putnedgrs	Store records in ecac_work which has identity field orid.
Usr8	pro_update_ecac	Update_ecac.sql	Remove remaining duplicate records which happen to have same lastname+emailaddr+putnedgrs	

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
Usr9	pro_update_ecac	Update_ecac.sql	Reset createdate to min(receivetim) from ecacrqst_ecacorder_min	.
Usr10	pro_update_users	Update_users.sql	Insert new records into users table from users_work	
Usr11	pro_update_users	Update_users.sql	Update users table with data from users_work	Update firstdate only if it is null.
Usr12	pro_update_users	Update_users.sql	Insert new records into users table from ecac_work	
Usr13	pro_update_users	Update_users.sql	Insert new records into users table for gsfcv0 data from scrs_users_work	Get emailaddr, calcaffili and firstdate from scrs_users_work. Set putnedgrs to current time and createdate to default time.

## FoxPro Database

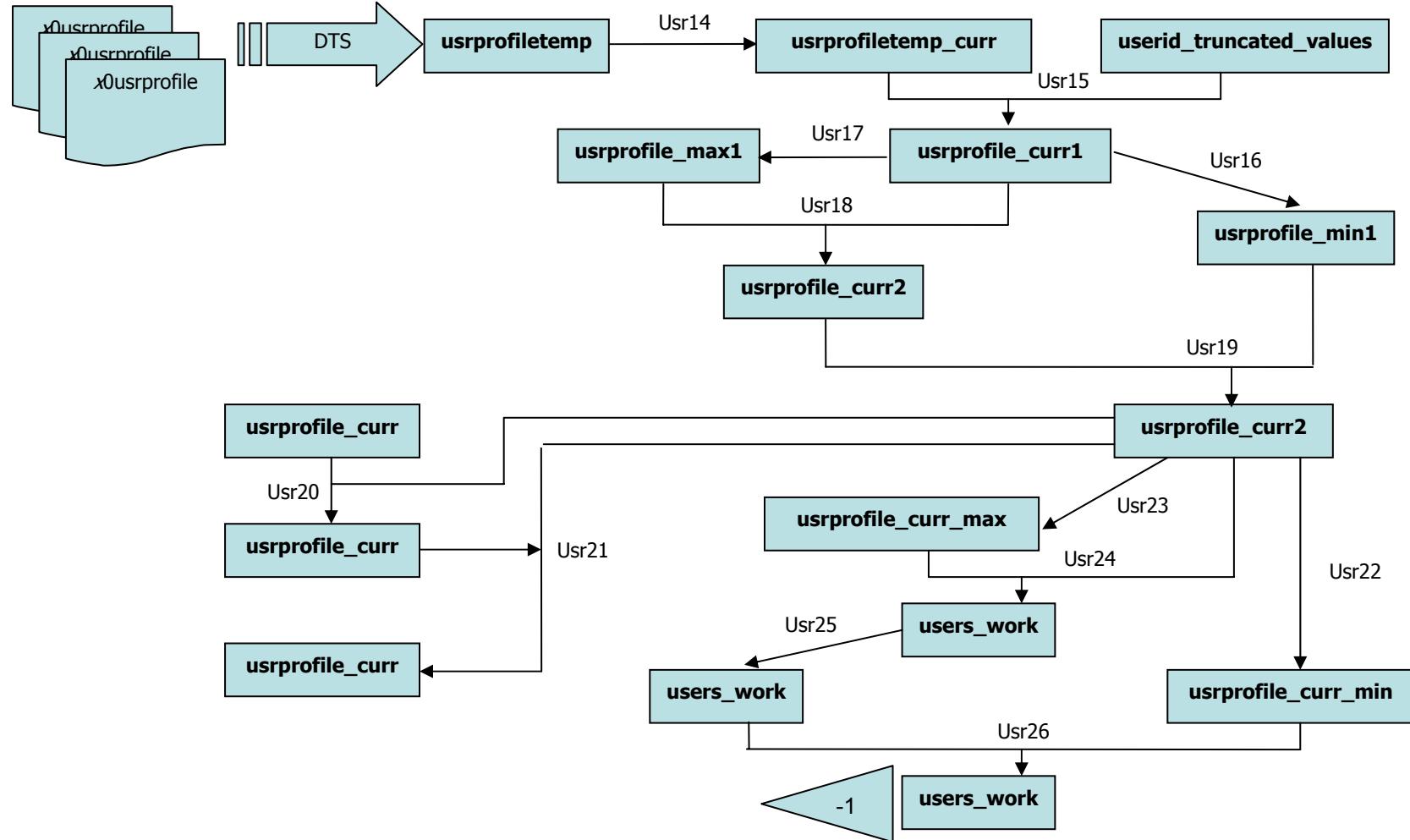


Figure 5-21 `usrprofile` Users – Ingest DFD

**Table 5-46 usrprofile Users – Ingest DFD Key**

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
Usr14	pro_update_userprofile_curr2	Update_userprofile_curr2.sql	Copy all records	Capitalize firstname, lastname and convert emailaddr to lowercase.
Usr15*	pro_update_userprofile_curr2	Update_userprofile_curr2.sql	Correct for truncation of userid	Derive calcaffili from emailaddr and firstdate from createdate
Usr16	pro_update_userprofile_curr2	Update_userprofile_curr2.sql	Generate usrprofile_min1 auxiliary table to extract earliest unique records based on provider and userid	distinct provider, userid, min(createdate) from usrprofile_curr1 group by provider, userid.
Usr17	pro_update_userprofile_curr2	Update_userprofile_curr2.sql	Generate usrprofile_max1 auxiliary table to extract latest unique records based on provider and userid	distinct provider, userid, max(putnedgrs) from usrprofile_curr1 group by provider, userid.
Usr18	pro_update_userprofile_curr2	Update_userprofile_curr2.sql	Apply usrprofile_max1 to extract records based on provider + userid with latest putnedgrs	Store records in usrprofile_curr2. Derive calcaffili from emailaddr and firstdate from createdate.
Usr19	pro_update_userprofile_curr2	Update_userprofile_curr2.sql	Reset createdate to min(createdate) from usrprofile_min1	
Usr20	pro_update_userprofile_curr2	Update_userprofile_curr2.sql	Update existing records with new data from usrprofile_curr2	
Usr21	pro_update_userprofile_curr2	Update_userprofile_curr2.sql	Insert new records from usrprofile_curr2	
Usr22	pro_update_userprofile_curr2	Update_userprofile_curr2.sql	Generate usrprofile_curr_min auxiliary table to extract earliest unique records based on lastname and emailaddr	distinct lastname, emailaddr, min(createdate) from usrprofile_curr2 group by lastname,emailaddr.
Usr23	pro_update_userprofile_curr2	Update_userprofile_curr2.sql	Generate usrprofile_curr_max auxiliary table to extract latest unique records based on lastname and emailaddr	distinct lastname, emailaddr, max(createdate) from usrprofile_curr2 group by lastname,emailaddr.

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
Usr24	pro_update_userprofile_curr2	Update_userprofile_curr2.sql	Apply usrprofile_curr_max to extract records based on lastname + emailaddr with latest putnedgrs	Store records in users_work which has identity field edrsid.
Usr25	pro_update_userprofile_curr2	Update_userprofile_curr2.sql	Remove remaining duplicate records which happen to have same lastname+emailaddr+putnedgrs	
Usr26	pro_update_userprofile_curr2	Update_userprofile_curr2.sql	Reset createdate to min(createdate) from usrprofile_curr_min	

- Step may no longer be needed

**Table 5-47 Definitions for Tables Used to Build User Characterization Tables**

Table ID	EDGRS Table Name	Primary Key	EDGRS Table Description	Comments
1	ecacordertemp	provider orderid homedaac	staging area for ecacorder data transferred from FoxPro ·	The structure of this file mimics the structure of the input data.
2	ecacordertemp1		staging area for ecacorder data with the same structure as ecacordertemp.	
3	ecacrqsttemp	provider orderid homedaac requestid rqstdaac	staging area for ecacrqst data transferred from FoxPro ·	The structure of this file mimics the structure of the input data.
4	ecacrqsttemp1		staging area for ecacrqst data with the same structure as ecacrqsttemp.	
5	ecac_work		Intermediate table containing new user characterization info based on lastname +	Info is used to update users table for orders

<b>Table ID</b>	<b>EDGRS Table Name</b>	<b>Primary Key</b>	<b>EDGRS Table Description</b>	<b>Comments</b>
			emailaddr	
6	ecacrqst_ecacorder_work ecacrqst_ecacorder_min ecacrqst_ecacorder_max		Auxilliary tables used to build ecac_work	
7	users	lastname emailaddr	Table containing user characterization info for users of EDGRS Distribution System based on unique values of lastname + emailaddr	Contains identity field edgrsid that uniquely identifies a user in the EDGRS Distribution System
8	usrprofiletemp	provider userid homedaac firstname lastname	Staging area for new user characterization data received from DAACs via FoxPro	
9	usrprofile_curr	provider userid	Table containing user characterization data based solely on provider + userid	Used to link dist table to users for non-order distributions, e.g. subscriptions,
10	usrprofile_curr2		Intermediate table containing unique set of new users based on provider + userid	Derived from usrprofiletemp. Used to update usrprofile_curr table and derive users_work table
11	userid_truncated_values		Table containing new and old values of userid values that require correction	
12	usrprofiletemp_curr usrprofile_curr1 usrprofile_curr_min1 usrprofile_curr_max1		Temporary tables used to derive usrprofile_curr2 from usrprofiletemp	
13	users_work	edgrsid	Intermediate table containing new unique user characterization data based on provider + userid	Info is used to update users table for non-order distributions, e.g. subscriptions

*Table 5-48 Index Definitions for Tables Used to Build User Characterization Tables*

Index ID	EDGRS Index Name	Table	EDGRS Index Description	Clust-ered	Unique Index	Comments
1	PK_users	users	Primary Key	Yes	Yes	
2	IDX_users_edrsid	users	Indexed on edrsid	No	Yes	
3	PK_usrprofile_curr	usrprofile_curr	Primary Key	Yes	Yes	
4	PK_ecacordertemp	ecacordertemp	Primary Key	Yes	Yes	
5	PK_ecacrqsttemp	ecacrqsttemp	Primary Key	Yes	Yes	
6	PK_usrprofiletemp	usrprofiletemp	Primary Key	Yes	Yes	

*Table 5-49 User Characterization Table Definitions*

Table ID	Table Name	Column Name	Column Type	Key	Allow Nulls	Description
	(TBS)					

### 5.5.7 Building the EDGRS ecac Tables

The ecac tables ecacorder and ecacrqst provide information on user orders and requests, respectively.

Notes:

- All procedures and tables described below are in the EdgrsCurrent database except where indicated otherwise.

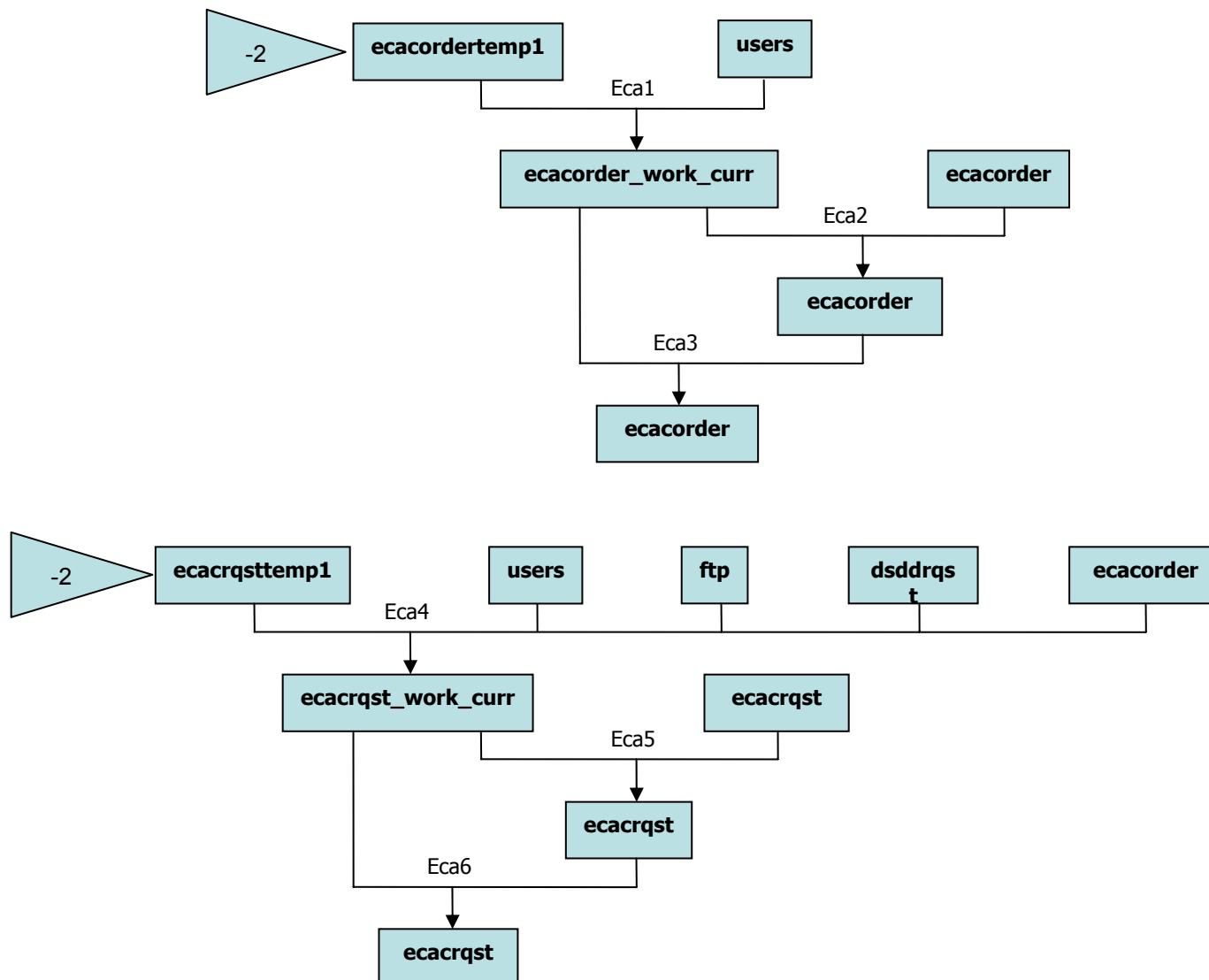


Figure 5-22 ecac Orders – Ingest DFD

**Table 5-50 ecac Orders – Ingest DFD Key**

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
Eca1	pro_normaliz_ecacorder_curr2	normaliz_ecacorder_curr2.sql	Generate intermediate table ecacorder_work_curr2 with edrsid from users table	
Eca2	pro_normaliz_ecacorder_curr2	normaliz_ecacorder_curr2.sql	Update existing records in ecacorder table with new data	
Eca3	pro_normaliz_ecacorder_curr2	normaliz_ecacorder_curr2.sql	Insert new records into ecacorder table	
Eca4	pro_normaliz_ecacrqst_curr2	normaliz_ecacrqst_curr2.sql	Generate intermediate table ecacrqst_work_curr2 with edrsid from users table	
Eca5	pro_normaliz_ecacrqst_curr2	normaliz_ecacrqst_curr2.sql	Update existing records in ecacrqst table with new data	
Eca6	pro_normaliz_ecacrqst_curr2	normaliz_ecacrqst_curr2.sql	Insert new records into ecacrqst table	.

**Table 5-51 Definitions for Tables Used to Build ecac Tables**

Table ID	EDGRS Table Name	Primary Key	EDGRS Table Description	Comments
1	ecacordertemp	provider orderid homedaac	staging area for ecacorder data transferred from FoxPro .	The structure of this file mimics the structure of the input data.
2	ecacordertemp1		staging area for ecacorder data with the same structure as ecacordertemp.	

Table ID	EDGRS Table Name	Primary Key	EDGRS Table Description		Comments
3	ecacrqsttemp	provider orderid homedaac requestid rqstdaac	staging area for ecacrqst data transferred from FoxPro		The structure of this file mimics the structure of the input data.
4	ecacrqsttemp1		staging area for ecacrqst data with the same structure as ecacrqsttemp.		
5	ecacorder	provider orderid	Table containing information about user orders		
6	ecacorder_work_curr		Intermediate table used to build ecacorder		
7	ecacrqst	provider orderid requestid	Table containing information about user orders		
8	ecacrqst_work_curr		Intermediate table used to build ecacrqst		

*Table 5-52 Index Definitions for Tables Used to Build ecac Tables*

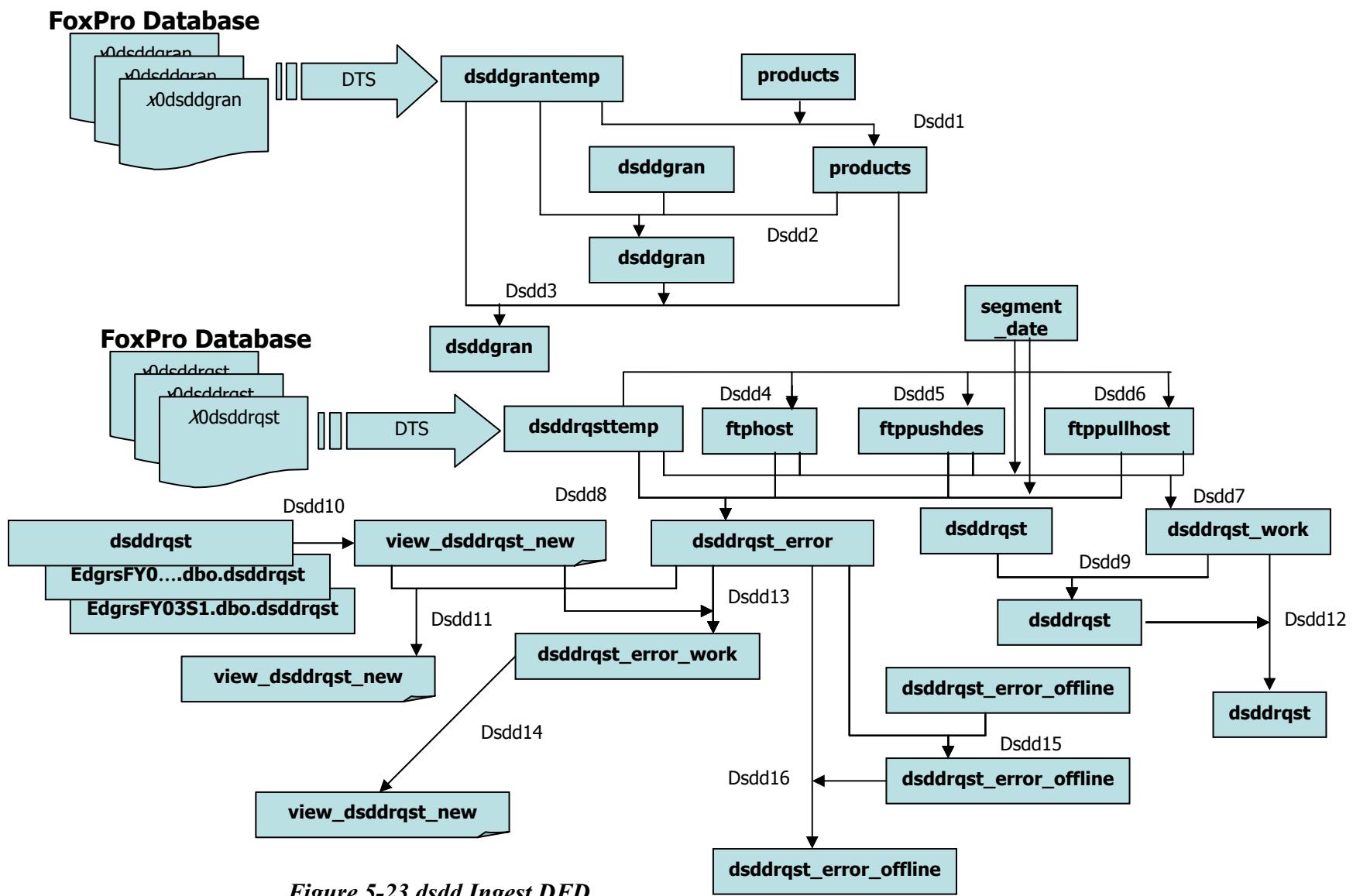
Index ID	EDGRS Index Name	Table	EDGRS Index Description	Clust-ered	Unique Index	Comments
1	PK_ecacorder_curr1	ecacorder	Primary Key	Yes	Yes	
2	PK_ecacrqstr_curr1	ecacrqst	Primary Key	No	Yes	
3	PK_ecacordertemp	ecacordertemp	Primary Key	Yes	Yes	
4	PK_ecacrqsttemp	ecacrqsttemp	Primary Key	Yes	Yes	

### 5.5.8 Building the EDGRS dsdd Tables

The dsdd tables dsddgran and dsddrqst provide volume and user information, respectively, on all data distribution requests, ECS and non-ECS.

Notes:

- All procedures and tables described below are in the EdgrsCurrent database except where indicated otherwise.



*Figure 5-23 dsdd Ingest DFD*

**Table 5-53 ecac Orders – Ingest DFD Key**

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
Dsdd1	pro_update_dsdggran	Update_dsdggran_curr.sql	Insert new records into products table	
Dsdd2	pro_update_dsdggran	Update_dsdggran_curr.sql	Update existing records in dsdggran table	
Dsdd3	pro_update_dsdggran	Update_dsdggran_curr.sql	Insert new records into dsdggran table	
Dsdd4	pro_normaliz_ftphost	Normaliz_dsdgrqst_curr.sql	Insert new records into ftphost table	
Dsdd5	pro_normaliz_ftppushdes	Normaliz_dsdgrqst_curr.sql	Insert new records into ftppushdes table	
Dsdd6	pro_normaliz_ftppullhos	Normaliz_dsdgrqst_curr.sql	Insert new records into ftppullhos table	.
Dsdd7	pro_normaliz_dsdgrqst	Normaliz_dsdgrqst_curr.sql	Generate intermediate dsdgrqst_work table from dsdgrqsttemp	Uses segdate in segment_date table to limit records to current dsdgrqst segment
Dsdd8	pro_normaliz_dsdgrqst	Normaliz_dsdgrqst_curr.sql	Put earlier records into dsdgrqst_error table	Uses segdate in segment_date table to limit records to data earlier than current segment
Dsdd9	pro_normaliz_dsdgrqst	Normaliz_dsdgrqst_curr.sql	Update existing records in current segment dsdgrqst	Uses dsdgrqst_work
Dsdd10	view_dsdgrqst_new	Create_dsdg_views.sql	View represents online part of dsdgrqst table	
Dsdd11*	pro_normaliz_dsdgrqst	Normaliz_dsdgrqst_curr.sql	Update existing records in online segments other than the current segment for the dsdgrqst table	Uses dsdgrqst_error table

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
Dsdd12	pro_normaliz_dsddrqst	Normaliz_dsddrqst_curr.sql	Insert new records into current dsddrqst segment	Uses dsddrqst_work
Dsdd13*	pro_normaliz_dsddrqst	Normaliz_dsddrqst_curr.sql	Put new earlier records into dsddrqst_error_work	Uses dsddrqst_error
Dsdd14	pro_normaliz_dsddrqst	Normaliz_dsddrqst_curr.sql	Insert new records into online segments other than the current segment for the dsddrqst table	Uses dsddrqst_error_work
Dsdd15*	pro_normaliz_dsddrqst	Normaliz_dsddrqst_curr.sql	Update existing offline records in dsddrqst_error_offline table	
Dsdd16*	pro_normaliz_dsddrqst	Normaliz_dsddrqst_curr.sql	Insert new offline records into dsddrqst_error_offline table	

\* Threshold time is hard-coded. Need to use offlinedate in segment\_date table instead.

**Table 5-54 Definitions for Tables Used to Build dsdd Tables**

Table ID	EDGRS Table Name	Primary Key/Identity	EDGRS Table Description	Comments
1	dsddgrantemp	provider requestid granuleid	staging area for dsddgran data transferred from FoxPro -	The structure of this file mimics the structure of the input data.
2	dsddgran	provider requestid granuleid starttime	Table containing information about distribution volume for the current segment of the dsddgran table.	
3	products	prodid	Table containing distinct products distributed	
4	dsddrqsttemp	provider requestid	staging area for dsddrqst data transferred from FoxPro -	The structure of this file mimics the structure of the input data.

<b>Table ID</b>	<b>EDGRS Table Name</b>	<b>Primary Key/ Identity</b>	<b>EDGRS Table Description</b>	<b>Comments</b>
5	dsddrqst	provider requestid starttime	Table containing information about distribution requests for the current segment of the dsddrqst table.	
6	ftphost	hostid	Table containing names of hosts to connect to for FTP push	
7	ftppushdes	pullid	Table containing directories of target systems for FTP pushes	
8	ftppullhos	pushid	Table containing names of hosts doing ftp pulling	
9	dsddrqst_error		Work table containing input records earlier than those in the current dsddrqst segment	
10	dsddrqst_error_offline		Intermediate table holding online records previously received that are earlier than those in the current segment	
11	dsddrqst_work dsddrqst_error_work		Work tables	

**Table 5-55 Index Definitions for Tables Used to Build dsdd Tables**

<b>Index ID</b>	<b>EDGRS Index Name</b>	<b>Table</b>	<b>EDGRS Index Description</b>	<b>Clust- ered</b>	<b>Unique Index</b>	<b>Comments</b>
1	PK_dsdgran01 PK_dsdgran_1	dsdgran Edgrsxx, dbo.dsdgran	Primary Key	Yes	Yes	
2	PK_dsdgrqst01 PK_dsdgrqst	dsdgrqst Edgrsxx, dbo.dsddrqst	Primary Key	No	Yes	
3	PK_dsdgrantemp	dsdgrantemp	Primary Key	Yes	Yes	
4	PK_dsdgrqsttemp	dsdgrqsttemp	Primary Key	Yes	Yes	

### 5.5.9 Ingesting the EDGRS Daily gsfcv0 Data

The daily gsfcv0 input data is ingested into the users, medias, scrs\_sid, dist and dsddgran tables.

Notes:

- All procedures and tables described below are in the EdgrsCurrent database except where indicated otherwise.

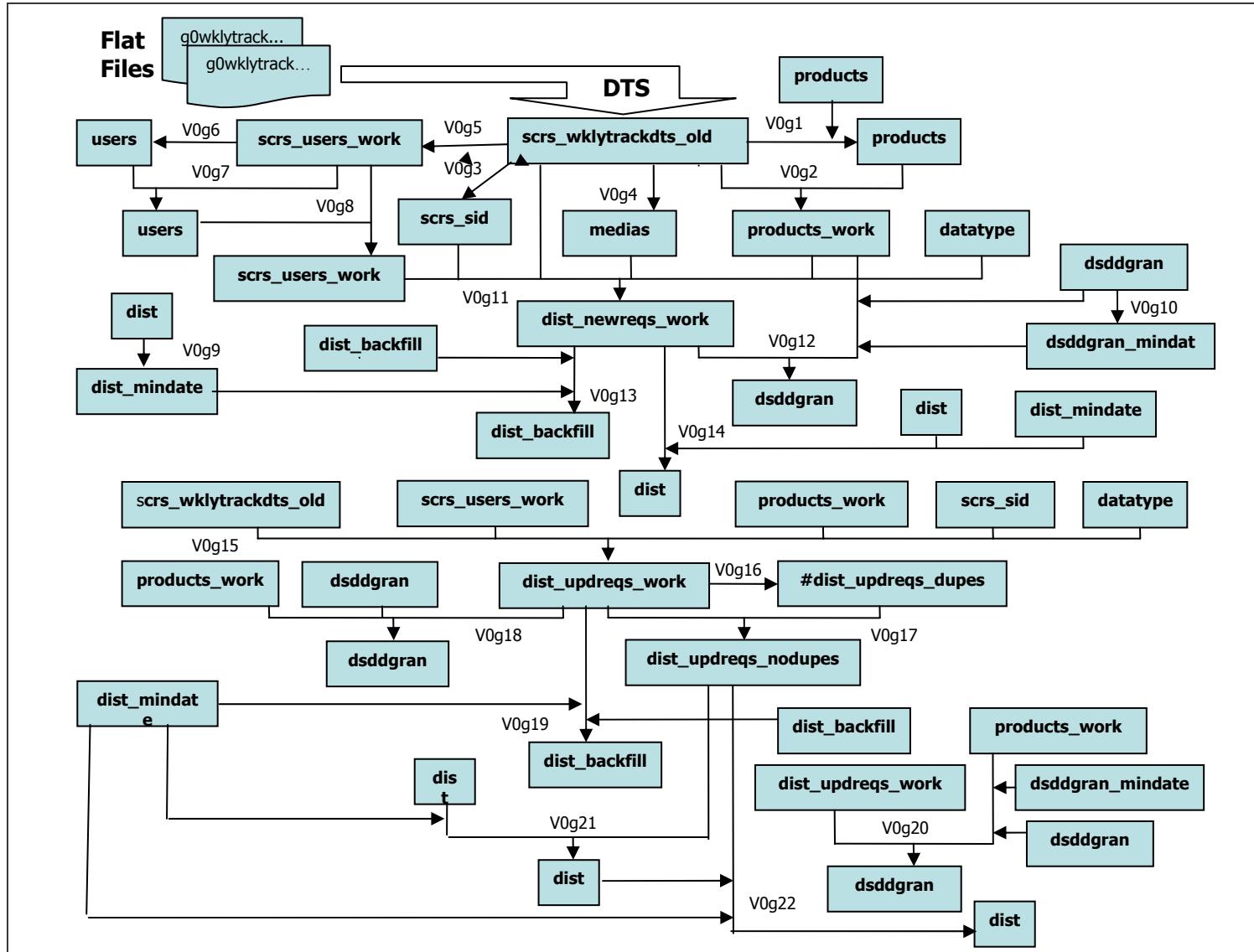


Figure 5-24 `gsfcv0` Ingest DFD

**Table 5-56 gsfcv0 Ingest DFD Key**

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
V0g1	pro_tracking_to_dist_curr1	Update_scrs_tracking_current1.sql	Insert new records into products table	
V0g2	pro_tracking_to_dist_curr1	Update_scrs_tracking_current1.sql	Extract all distinct products from scrs_wklytrackdts_old into products_work table	
V0g3	pro_tracking_to_dist_curr1	Update_scrs_tracking_current1.sql	Insert new records into scrs_sid table	
V0g4	pro_tracking_to_dist_curr1	Update_scrs_tracking_current1.sql	Insert new records into medias table	For requested products and delivered products
V0g5	pro_tracking_to_dist_curr1	Update_scrs_tracking_current1.sql	Generate intermediate scrs_users_work table of distinct users from scrs_wklytrackdts_old	Get emailaddr form input data, calcaffili from emailaddr . Set daac to gsfcv0, edgrsid to null. Set firstdate using get_firstdate_trk
V0g6	pro_tracking_to_dist_curr1	Update_scrs_tracking_current1.sql	Insert new records into users table with emailaddr, calcaffili, firstdate from scrs_users_work table	Set emailaddr to lowercase and lastname to " (lastname not provided)
V0g7	pro_tracking_to_dist_curr1	Update_scrs_tracking_current1.sql	Update firsdate and calcaffili of existing records in users table	If new date is earlier than current Firstdate
V0g8	pro_tracking_to_dist_curr1	Update_scrs_tracking_current1.sql	Use users table to backfill edgrsid in scrs_users_work table	
V0g9	pro_tracking_to_dist_curr1	Update_scrs_tracking_current1.sql	Generate dist_mindate from current dist segment	Uses starttime
V0g10	pro_tracking_to_dist_curr1	Update_scrs_tracking_current1.sql	Generate dsddgran_mindate from current dsddgran segment	Uses starttime
V0g11*	pro_tracking_to_dist_curr1	Update_scrs_tracking_current1.sql	Generate intermediate table dist_newreqs_work containing info from new gsfcv0 records requested	Where prodreqd='1'. Sets orderid based on prodreqid, requestid based on prodreqid and productid, Sets shipdate, endtime based on get_lastdate_trk and orderdate, starttime based on get_lastdate_trk

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
V0g12	pro_tracking_to_dist_curr1	Update_scrs_tracking_current1.sql	Insert new records into current dsddgran segment	Uses dist_newreqs_work table
V0g13*	pro_tracking_to_dist_curr1	Update_scrs_tracking_current1.sql	Put records earlier than the current segment into dist_backfill	Uses dist_newreqs_work
V0g14	pro_tracking_to_dist_curr1	Update_scrs_tracking_current1.sql	Insert new gsfcv0 records into current dist segment from dist_newreqs_work	
V0g15*	pro_tracking_to_dist_curr1	Update_scrs_tracking_current1.sql	Generate intermediate table dist_updreqs_work containing info for existing gsfcv0 records completed, closed or cancelled.	Where completed='1' or closed='1' or cancelled='1' or comp='Cancelled' or comp='Completed'. Sets orderid based on prodreqid, requestid based on prodreqid and productid, Sets shipdate, endtime based on get_lastdate_trk and orderdate, starttime based on get_lastdate_trk Derives calcaffili from emailaddr
V0g16*	pro_tracking_to_dist_curr1	Update_scrs_tracking_current1.sql	Generate temporary table that contains all records including dupes  when grouped by provider, requestid, calcesdt and starttime	Includes max(endtime) and min(endtime)  Used to eliminate duplicates when updating or inserting records into current dist segment
V0g17	pro_tracking_to_dist_curr1	Update_scrs_tracking_current1.sql	Extract records to keep from temporary table and store in dist_updreqs_nodupes	Uses dist_updreqs_work and temporary table #dist_updreqs_dupes
V0g18	pro_tracking_to_dist_curr1	Update_scrs_tracking_current1.sql	Update status for existing records in current dsddgran segment	Uses dist_updreqs_work
V0g19	pro_tracking_to_dist_curr1	Update_scrs_tracking_current1.sql	Put records earlier than the current dist segment into dist_backfill	Uses dist_updreqs_work
V0g20	pro_tracking_to_dist_curr1	Update_scrs_tracking_current1.sql	Put new records into current dsddgran segment	Uses dist_updreqs_work

ID	Procedure/View	File Defining Procedure/View	Action / Key for Relation	Comments
V0g21	pro_tracking_to_dist_curr1	Update_scrs_tracking_current1.sql	Update existing records in current dist segment	Uses dist_updreqs_nodupes
V0g22	pro_tracking_to_dist_curr1	Update_scrs_tracking_current1.sql	Insert new records into current dist segment	Uses dist_updreqs_nodupes

\* Threshold time is hard-coded. Need to use offlinedate in segment\_date table instead.

*Table 5-57 Definitions for Tables Used to Ingest the gsfcv0 Data*

Table ID	EDGRS Table Name	Primary Key/ Identity	EDGRS Table Description	Comments
1	scrs_wklytrackdts_old		staging area for gsfcv0 data transferred from flat files .	The structure of this file mimics the structure of the input data.
2	medias	media_txt	Table used to normalize the various textual media names into a common media name	
3	products	prodid	Table containing distinct products distributed	
4	scrs_sid	source_name source_txt or sid	Table used to normalize different data sources	
5	users	lastname emailaddr	Table used to record distinct users in EDGRS.	
6	datatype	datatypeid	Table used to uniquely map the ESDT names to mission, instrument, level, and discipline for reporting purposes	
7	scrs_users_work products_work dist_newreqs_work		Work tables used in processing gsfcv0 data	

Table ID	EDGRS Table Name	Primary Key/Identity	EDGRS Table Description	Comments
	dist_updreqs_work dist_updreqs_nodupes dist_mindate dsddgran_mindate			
8	dsddgran		Table containing information about distribution volume	
9	dist	provider requestid calcesdt starttime	Primary source of metrics on data distributed from EOSDIS sites	

*Table 5-58 Index Definitions for Tables Used to Ingest the gsfcv0 Data*

Index ID	EDGRS Index Name	Table	EDGRS Index Description	Clust-ered	Unique Index	Comments
1	PK_dsddgran01	dsddgran	Primary Key	Yes	Yes	
2	PK_dist_2	dist	Primary Key	No	Yes	
3	PK_dsddgrantemp	dsddgrantemp	Primary Key	Yes	Yes	
4	PK_dsdrrqsttemp	dsdrrqsttemp	Primary Key	Yes	Yes	
5	INDX_sortendtime	dist	Index on sortendtime	No	No	
6	PK_users	users	Primary Key	Yes	Yes	
7	INDX_users_edgrsid	users	Index on edgrsid	No	Yes	

5.5.10 Building the OM Tables(TBS)

5.5.11 Building Other Supporting Tables (TBS)

## **6 Process Control (TBS)**

## Appendix A EDGRS Report Definition Table

(Update and expand-TBS)

This appendix defines the detailed attributes for the reports that have been generated for the EDGRS system.

The report naming convention was altered to better keep track of the data source for that report. It is summarized as follows:

rINnnn	Report (R) or Graph (G) for INGEST (IN) with sequence number nnn (e.g. 001)
rARnnn	Report (R) or Graph (G) for ARCHIVE (AR) with sequence number nnn (e.g. 001)
rPRnnn	Report (R) or Graph (G) for PRODUCTION (PR) with sequence number nnn (e.g. 001)
rDFnnn	Report (R) or Graph (G) for DISTRIBUTE FROM ARCHIVE (DF) with sequence number nnn (e.g. 001)
rDUnnn	Report (R) or Graph (G) for DISTRIBUTE TO USERS (DU) with sequence number nnn (e.g. 001)
rXXnnn	Report (R) or Graph (G) for OTHER Special Purposes (XX) with sequence number nnn (e.g. 001)

'r' may be either 'R' for Report or 'G' for Graph. The report names should appear in the footer of each report. The old report names are indicated since either name may be used. Reports with an old report name beginning with a '%' are not currently active reports.

New Report Name	Old Report Name	Primary Use	EDGRS Tables (Sql Server Summary Table)	DAAC	Report Title	Constraining Fields Used	Other Fields Used	Other Functions
<b>Ingest Reports</b>								
RIN001	monsumm1 / sum1ings	Web matrix	inreqdata,datatype d (rp_ingest_summary)	Multi	Data Ingested By Instrument	datatype=d.datatypeid; state='Archived' or 'Successful'; a>=procend<=b; not empty(requestid)	d.edginstrum, volume	
RIN002	mon411r	Web matrix	inreqdata,datatype (rp_ingest_instrument_datatype)	Multi	Data Ingested By Instrument	state='Archived' or 'Successful'; a>=procend<=b;	datatype, volume, edginstrum, platform	get_instru, get_pltfrm
RIN003	mon43r	Web matrix	inreqdata (rp_ingest_instrument_datatype)	Multi	Data Ingested By Datatype	state='Archived' or 'Successful'; a>=procend<=b	datatype, volume	

New Report Name	Old Report Name	Primary Use	EDGRS Tables (Sql Server Summary Table)	DAAC	Report Title	Constraining Fields Used	Other Fields Used	Other Functions
RIN004	N/A	SPSO	inreqhdr h, inreqdata d, datatype t, programs p	Single	Ingest Detail (for SPSO)	not empty(d.requestid) or not empty(d.requestid); d.datatype=t.datatypeid, h.requestid=d.requestid; d.state='Archived' or 'Successful'; a>=h.procendtim<=b	d.datatype, d.volume, t.mission, t.edginstrum, p.datasource, h.extdatapro	
RIN005	N/A	Web	inreqdata,datatype (rp_ingest_instrument_datatype)	Multi	Ingest by Provider – Summary	d.datatype=t.datatypeid, h.requestid=d.requestid; d.state='Archived' or 'Successful'; a>=h.procendtim<=b	TBS	
RIN006	N/A	Web	inreqdata,datatype (rp_ingest_instrument_datatype)	Multi	Ingest by Provider – Detail	d.datatype=t.datatypeid, h.requestid=d.requestid; d.state='Archived' or 'Successful'; a>=h.procendtim<=b	TBS	
RIN007	N/A	Web	inreqdata,datatype (rp_ingest_instrument_datatype)	Multi	Ingest by Provider – Daily	d.datatype=t.datatypeid, h.requestid=d.requestid; d.state='Archived' or 'Successful'; a>=h.procendtim<=b	TBS	
<b>Archive Reports</b>								
RAR001	monsummarch/sumarchs	Web matrix	dsmdgran,datatype d (rp_archive_summary)	Multi	Data Archived By Instrument	not empty(dbid); a>=inserttime<=b; shortname=d.datatypeid	d.edginstrum, sizembecs	
RAR002	mon412r	Web matrix	dsmdgran,datatype d (rp_archive_instrument_datatype)	Multi	Data Archived By Instrument	not empty(dbid); a>=inserttime<=b; shortname=d.datatypeid	edginstrum, d.mission, sizembecs	get_instru, get_pltfrm
RAR003	mon420r	Web matrix	Dsmdgran (rp_archive_instrument_datatype)	Multi	Data Archived By Data Type	not empty(dbid); a>=inserttime<=b	shortname, sizembecs	
RAR005	N/A	SPSO	dsmdgran, datatype d, programs p	Single	Archive Detail (for SPSO) by processing date	not empty(dbid); shortname=d.datatypeid; a>=inserttime<=b	sizembecs, d.edgrinstru, d.mission, p.datasource	
RAR005d	N/A	SPSO	dsmdgran	Single	Archive Detail (for SPSO) by data date	not empty(dbid); shortname=d.datatypeid; a>=begintime<=b	sizembecs, d.edgrinstru, d.mission, p.datasource	
Rar006	N/A	Web	Dsmdgran (rp_archive_instrument_datatype)	Multi	Archive by DAAC Summary	shortname=d.datatypeid; a>=inserttime<=b	TBS	
RAR007	N/A	Web	Dsmdgran (rp_archive_instrument_datatype)	Multi	Archive by DAAC Detail	shortname=d.datatypeid; a>=inserttime<=b	TBS	
RAR008	N/A	Web	Dsmdgran (rp_archive_instrument_datatype)	Multi	Archive by DAAC Daily	shortname=d.datatypeid; a>=inserttime<=b	TBS	

New Report Name	Old Report Name	Primary Use	EDGRS Tables (Sql Server Summary Table)	DAAC	Report Title	Constraining Fields Used	Other Fields Used	Other Functions
RAR009	N/A	Web	Dsmdgran (rp_archive_instrument_datatype)	Multi	Archive by Data Time Summary	shortname=d.datatypeid; a>=inserttime<=b	TBS	
RAR010	N/A	Web	Dsmdgran (rp_archive_instrument_datatype)	Multi	Archive by Data Time Detail	shortname=d.datatypeid; a>=inserttime<=b	TBS	
RAR011	N/A	Web	Dsmdgran (rp_archive_instrument_datatype)	Multi	Archive by Data Time Daily	shortname=d.datatypeid; a>=inserttime<=b	TBS	
<b>Production Reports</b>								
RPR001	monsummprod/sumprods	Web matrix	pldgshort,datatype d (TBS)	Multi	Data Processed By Instrument	not empty(granuleid); a>=timestamp<=b; datatypeid=d.datatypeid	d.edgrinstrum	
RPR002	mon413r	Web matrix	pldgshort,datatype (TBS)	Multi	Data Processed By Instrument	not empty(granuleid); a>=timestamp<=b; datatypeid=d.datatypeid	d.edgrinstrum, d.mission	get_instr; get_pltfrm
RPR003	mon417r	Web matrix	Pldgshort (TBS)	Multi	Data Processed By Datatype	not empty(granuleid); a>=timestamp<=b	datatypeid	
RPR004	mon414r	On Demand	dprs	Multi	Data Processing Request Completion Times By Platform	not empty(dprid); a>=completion<=b	pgecpertime, pgeelapsed, platform	
RPR005	mon415r	On Demand	dprs	Multi	Data Processing Request Completion Times By DPRTType	not empty(dprid); a>=completion<=b	pgecpertime, pgeelapsed, platform	
RPR006	N/A	SPSO		Single	Data Processing Detail (for SPSO)			
<b>Distribute From Archive Reports</b>								
RDF001	monsummfromarc/sumfromarc	Web matrix	dsddrqst,datatype d	Multi	Data Retrieved From Archived By Instrument	a>=endtime<=b; state='Shipped'; esdtype=d.datatypeid	sizeinmb/1048576, d.edgrinstrum	get_instr
RDF002	mon418r	Web matrix	dsddrqst,datatype d (rp_dfa_instrument_datatype)	Multi	Data Retrieved From Archive By Instrument	a>=endtime<=b; state='Shipped'; esdtype=d.datatypeid	sizeinmb/1048576, d.edgrinstrum, d.mission	get_instr, get_pltfrm
RDF003	mon419r	Web matrix	Dsddrqst, datatype d (rp_dfa_instrument_datatype)	Multi	Data Retrieved From Archive By DataType	a>=endtime<=b; state='Shipped'	sizeinmb/1048576, esdtype, mediatype	
RDF004	mon418rb	On Demand	dsddrqst, datatype d, subscrips	Multi	Data Retrieved From Archive By Instrument and Usertype	a>=endtime<=b; state='Shipped'; esdtype=d.datatypeid; ecsuserid=s.userid; orderid<>'NULL' and not empty(orderid)	sizeinmb/1048576	get_instr, get_usertype
RDF005	mon41r	On Demand	dsddrqst	Multi	Total Data Retrieved From Archive By MediaType	a>=endtime<=b; state='Shipped'	sizeinmb/1048576, mediatype, orderid, ecsuserid, nrgranules	

New Report Name	Old Report Name	Primary Use	EDGRS Tables (Sql Server Summary Table)	DAAC	Report Title	Constraining Fields Used	Other Fields Used	Other Functions
RDF006	mon41ra	On Demand	dsddrqst, subscrips	Multi	Total Data Retrieved From Archive By MediaType and Usertype	a>=endtime<=b; state='Shipped'; ecsuserid=s.userid; orderid<>'NULL' and not empty(orderid)	sizeinmb/1048576, mediatype, nrgranules	get_usertype
RDF007	mon421r	M&O	dsddrqst, subscrips	Multi	Total Data Retrieved From Archive By UserType By MediaType	a>=endtime<=b; state='Shipped'; ecsuserid=s.userid; orderid<>'NULL' and not empty(orderid)	sizeinmb/1048576, mediatype, nrgranules	get_usertype
RDF008	N/A	SPSO		Single	Data Distributed by Subscription (for SPSO)			
RDF009	N/A	M&O (Generates KP1, KP2, KP3 delimited files)	dsddrqst	Multi	Data Distributed by Subscription and SCF (for kprickett)			
RDF010	N/A	SPSO	dsddrqst	Single	Data Distributed by Subscription and SCF (for SPSO)			
RDF011	N/A	SPSO	dsddrqst	Single	Data distributed from archive for external users			
RDF012	N/A	Web	dsddrqst, datatype d, subscrips (rp_dfa_instrument_datatype)	Multi	Data Retrieved From Archive by Subscription Summary	Usertype=1 (known subscription)		
RDF013	N/A	Web	dsddrqst, datatype d, subscrips (rp_dfa_instrument_datatype)	Multi	Data Retrieved From Archive by Subscription Daily	Usertype=1 (known subscription)		
<b>Distribute to External User Reports</b>								
RDU001	monsummdist/sumdist	Web matrix	ecacrqst	Multi	Distribution to Users	Status='Shipped'; a>=calctime<=b	orderid, requestid, numgranule, numfiles, numbytes	get_instru, get_usertype
RDU002	mondistinst/sumdistinst	Web matrix	programs p, ecacrqst r, eacorder o, dsddrqst d, subscrips, datatype t	Multi	Distribution to Users by Instrument/ESDT	r.Status='Shipped'; a>=calctime<=b; r.orderid=d.orderid; r.requestid=d.requestid; d.ecsuserid=s.userid; r.orderid<>'NULL' and not empty(orderid); r.orderid=o.orderid; d.esdttype=t.datatypeid	r.numgranule, r.numfiles, r.numbytes, d.mediatype, dstarttime	get_usertype, get_instru
RDU003	mon45r	On Demand	ecacrqst	Multi	User Request	Status='Shipped';	shiptime-calctime	

New Report Name	Old Report Name	Primary Use	EDGRS Tables (Sql Server Summary Table)	DAAC	Report Title	Constraining Fields Used	Other Fields Used	Other Functions
					Turnaround	a>=calctime<=b		
RDU004	mon46r	On Demand	ecacrqst, reqstate r	Multi	User Request Distribution Status	Status=r.reqstate; a>=calctime<=b	r.reqgroup	
RDU005	RDU005	M&O	programs p, ecacrqst r, ecacorder o, dsddrqst d, subscr s	Single	Distribution to Users	r.Status='Shipped'; a>=calctime<=b; r.orderid=d.orderid; r.requestid=d.requestid; d.ecsuserid=s.userid; r.orderid<>'NULL' and not empty(orderid); r.orderid=o.orderid; d.esdtype=t.datatypeid	r.numgranule, r.numfiles, r.numbytes, d.mediatype, d.starttime	get_usertype
RDU006	mondistdt	Web matrix	programs p, ecacrqst r, ecacorder o, dsddrqst d, subscr s, datatype t	Multi	Distribution to Users by Provider/ESDT	r.Status='Shipped'; a>=calctime<=b; r.orderid=d.orderid; r.requestid=d.requestid; d.ecsuserid=s.userid; r.orderid<>'NULL' and not empty(orderid); r.orderid=o.orderid; d.esdtype=t.datatypeid	r.numgranule, r.numfiles, r.numbytes, d.mediatype, d.starttime	get_usertype, get_instru
RDU007	N/A	Web	Dist	Multi	Distribute to Users – By Instrument			
RDU008	N/A	Web	Dist	Multi	Distribute to Users – By Datatype			
RDU010g	N/A	Web Graph	dsddrqst r, dsddgrang, datatype d	Multi	Top Ten Datatypes Distributed To Users Across All DAACs	g.Status='Shipped'; a>=endtime<=b; g.esdtype='Multiple'; r.requestid=g.requestid; r.orderid<>'NULL' and not empty(r.orderid); (datatype)=d.datatypeid	g.granuleid, g.granulesiz, d.descriptio	
GDF001	N/A	Web Graph	dist, datatype d	Single	Volume of “instrument” Level “level” Data Distributed by “Provider”	a>=endtime<=b; provider = prefix; State='Shipped'; datatypeid=calcesdt; edginstrum=instrum; level=lev; usertype=2	bytes/1048576/1024 (i.e.GB),requestid, orderid	

New Report Name	Old Report Name	Primary Use	EDGRS Tables (Sql Server Summary Table)	DAAC	Report Title	Constraining Fields Used	Other Fields Used	Other Functions
GDF002	N/A	Web Graph	dist, datatype d	Single	Volume of "instrument" Level "level" "discipline" Data Distributed by "Provider"	a>=endtime<=b; provider = prefix; State='Shipped'; datatypeid=calcesdt; edginstrum=instrum; level=lev; usertype=2, discipline=disciplin	bytes/1048576/1024 (i.e.GB),requestid, orderid	
<b>Other Reports</b>								
RXX001	N/A	Web Report	datatype	N/A	Data Type Definition		datatype, edgrinstrum, mission, level, discipline	

## **Appendix B      Detailed Table Design**

**(Update and expand)**

The following tables provide table and field level descriptions for the environment defined to support the EDGRS processing.

Table B-1 provides a description of the tables used in both the ECS environment and the EDGRS environment and provides a mapping between them.

Column Title	Description
Table ID	Identifier to associate references to a specific ECS Table.
ECS Subsystem	The subsystem in ECS to which this table belongs. They are encoded as follows: DMS - Data Management Subsystem INGST - Ingest Subsystem IOS - Interoperability Subsystem PDPS - Planning and Data Processing Subsystem SDSRV - Science Data Server Subsystem STGMT - Storage Management and Data Distribution SUBSRV - Subscription Server MSS - Management Support Subsystem
ECS Table Name (or source)	Name of Table in ECS environment. Archive tables are the same structure as the parent table with the 'Archive' postscript. Archive tables contain deleted records from the parent table. In parentheses is the name of the script running in the ECS environment that pulls the data from this table. If the source of the data is not an ECS table, the name of the source is identified. E.g., Operations is from a human defined source controlled by the operations group.
EDGRS Table Name	Name of table in the EDGRS environment that contains data extracted by the EDGRS Insert Script. There is a column for the FoxPro tables and the Sql Tables.
Primary Key	EDGRS primary key. There is a column for the FoxPro tables and the Sql Tables.

EDGRS Table Description	Comments about the meaning of the records and any associations that are pertinent. There is a column for the FoxPro tables and the Sql Tables. FoxPro tables may have a prefix identifying the source of the data (e.g., g0 for gsfc).
-------------------------	--

Table B-2 provides a description of the fields in the ECS table that are imported into the EDGRS environment.

Column Title	Description
ECS Table ID	Identifier to associate references to a specific ECS Table.
ECS Script Column Name	Resulting column name as pulled from the ECS table by the ECS script defined in the table above.
ECS Script Column Length	Number of character positions for this field in the resulting text file as pulled from the ECS table by the ECS script defined in the table above.
EDGRS Table Name	Name of the EDGRS table containing this field. There is a column for the FoxPro tables and the Sql Tables.
EDGRS Column Name	Name of this field in the associated EDGRS table. There is a column for the FoxPro tables and the Sql Tables.
EDGRS Column Type/Length/Decimal	Data type of this field in the associated EDGRS table. The valid types are defined as follows: N - Numeric C - Character D - Date Dec – Decimal (SQL Server only) V – Varchar (SQL Server only) I – Integer Memo – Memo Includes number of total character positions allocated for this field. For numeric fields, this includes the decimal and fraction components. For date fields, it represents the space allocated by FoxPro for this value. For numeric fields it also includes the precision of the fraction part. For example, 2 means 2 decimal places to the right of the decimal. There is a column for the FoxPro tables and the Sql Tables.
Description	General information about what the field means.

**Table B-1 Table Level Descriptions**

Table ID	ECS Sub-system	ECS Table Name (or Source)	EDGRS Table Name (FoxPro)	Primary Key (FoxPro)	EDGRS Table Description (FoxPro)	EDGRS Table Name (Sql Server)	Primary Key (Sql Server)	EDGRS Table Description (SQL Server)
1	INGST	InRequestSummary Header (inreqhdr.scr)	inreqhdr	RequestID	One record for each ingest request whose ingest processing is completed. 1-to-many with Table 2.	inreqhdr	provider, requestid	One-to-one with FoxPro table inreqhdr with provider added
2	INGST	InRequestSummary Data (inreqdata.scr)	inreqdata	RequestID, DataGranuleID, DataType	One record for each granule in the ingest request. many-to-1 with Table 1. Records in the 'new' state are not retrieved since they do not have a ProcessingStartDateTim e date association yet.	inreqdata	provider, requestid, granuleid	One-to-one with FoxPro table inreqdata with provider added
3	SDSRV	DsMdGranules	dsmdgra n	dbID, shortname	One record for each granule archived.	dsmdgran	provider, dbid	One-to-one with FoxPro table dsmdgran with provider added
4	STGMT	DsDdRequestArchive, DsDdRequest	dsddrqst	RequestId	One record per distribution request	dsddrqst, dsddrqsttem p, ftphost, ftppullhos, ftppushdes	provider, requestid	dsddrqst is one-to-one with FoxPro table dsddrqst but normalized with an ftphostid, ftppushdestid and ftppullhosid; with provider added dsddrqstempt is one-to-one with FoxPro table dsddrqst with provider added
5	STGMT	DsDdParameterListArchive, DsDdParameterList	dsddrqst	RequestId	One record per distribution request	dsddrqst		

Table ID	ECS Sub-system	ECS Table Name (or Source)	EDGRS Table Name (FoxPro)	Primary Key (FoxPro)	EDGRS Table Description (FoxPro)	EDGRS Table Name (Sql Server)	Primary Key (Sql Server)	EDGRS Table Description (SQL Server)
6	STGMT	DsDdGranuleArchive, DsDdGranule	dsddgran	RequestId, granuleId	One record for each granule distributed.	dsddgran	provider, requestid ,	One-to-one with FoxPro table dsddgran with provider added
7	PDPS	PIDprCompletion, PIDprCompletion Archive	dprs	DprId, CompletionDate	One record per data processing request.	dprs	provider, dprid, completi on	One-to-one with FoxPro table dprs with provider added
8	PDPS	PlDataGranuleShort, PlDataGranuleShort Archive	pldgshort	GranuleId	One record per granule input or output from production processing.	pldgshort	provider, granuleid	One-to-one with FoxPro table pldgshort with provider added
9	MSS	EcAcRequest	ecacrqst	OrderId, RequestId, OrderHomeDAA C, RequestHomeDA AC	One record per user distribution request	ecacrqst ftp	provider, requestid	One-to-one with FoxPro table ecacrqst with provider added and user info normalized (i.e. EDGRSID points to all user info kept in usrprofile). USERID also kept from ecacorder.
10	MSS	EcAcOrder	ecacorder	OrderId , OrderHomeDAA C	One order record can be 1-to-many request records (Table 10). One record per user order.	ecacorder	provider, orderid	One-to-one with FoxPro table ecacorder with provider added and user info normalized, i.e. EDGRSID points to all user info kept in usrprofile.
11	MSS	MsAcUsrProfile	usrprofil e	HomeDAAC, Userid, LastName, FirstName	One record per registered user	usrprofile_c urr	edgrsid, lastname, emailadd r	User table normalized to include all registered users from usrprofile plus all users requesting data. All unique user data kept here and a unique EDGRSID is assigned to each unique user

Table ID	ECS Sub-system	ECS Table Name (or Source)	EDGRS Table Name (FoxPro)	Primary Key (FoxPro)	EDGRS Table Description (FoxPro)	EDGRS Table Name (Sql Server)	Primary Key (Sql Server)	EDGRS Table Description (SQL Server)
								(LASTNAME+EMAILADDR). Note PROVIDER is not a key.
12	PDPS	PlDataProcessing Request	pldgreq	dprid	One record per data processing request (DPR)			
13	PDPS	PlDprData (&PlDprCompletion)	pldprdat a	dprid	One record per granule in a DPR			
14	N/A	Operations	Subscrip	userid, scf	List of subscriptions to external SIPS or groups for further processing or archival	subscrip	provider, userid	One record per unique subscription
15	N/A	Operations or document 910-TDA-019-Rev12.xls	Datatypr e	Datatypeid	A unique datatype definition with mappings to instrument, mission, discipline, level, etc.	datatype	datatypeid	One record per unique datatype
16		dsddrqst				ftphost	hostid	One record per unique ftphost
17		dsddrqst				ftppullhos	pullid	One record per ftppull host address
18		dsddrqst				ftppushdes	pushid	One record per ftppush destination
19		ecacrqst				ftp	ftpid	One record per unique ftpaddress
20	N/A	EDGRS	Affiliati on	Upper(affil)	Definition of the affiliation value codes that can be assigned to a user			
21	N/A	EDGRS	ActionLog	datestamp	Output log of pull script execution			
22	N/A	EDGRS	Err	dtoc(date)+time	Log of errors during execution of			

Table ID	ECS Sub-system	ECS Table Name (or Source)	EDGRS Table Name (FoxPro)	Primary Key (FoxPro)	EDGRS Table Description (FoxPro)	EDGRS Table Name (Sql Server)	Primary Key (Sql Server)	EDGRS Table Description (SQL Server)
					EDGRS.exe (data importing into FoxPro and report generation)			
23	N/A	EDGRS	QALog2	primarykey	Output log of execution of EDGRS.exe			
24	N/A	EDGRS	Tables	str(id)+str(sortfield)	Driver for the execution of EDGRS.exe			
25	DataPool	DlGranuleAccess, DlFile, DlGranules	dpgranule	Dbid+accessType+fileType+ttoc(accessTime,1)	One record per dbid/granuleid, accessType, , fileType and accessTime from distributed from the DataPool	dpgranule	Provider, dbid, accessType, fileType, accessTime	Data Pool Granule (Access) table
26	MSS	Usrprofile,ecacrqs,ecacorder				users	Lastname, emailaddr	One record per unique lastname/emailaddr

**Table B-2 Field Level Descriptions**

ECS Table ID	ECS (Script) Column Name	ECS (Script) Column Length	EDGRS (FoxPro) Table Name	EDGRS (FoxPro) Column Name	EDGRS (FoxPro) Column Type	EDGRS (SQL Server) Table Name	EDGRS (SQL Server) Column Name	EDGRS (SQL Server) Column Type	Description
					inreqhdr	provider	V10		Provider/DAAC originating this record
1	RequestID	11	inreqhdr	requestid	C11	inreqhdr	requestid	V11	Request identifier automatically generated from InNextAvailableId table.
1	ExternalDataProvider	20	Inreqhdr/inreqdata	extdatapro	C20	inreqhdr	extdatapro	V20	Name of External Data Provider
1	Mission	60	inreqhdr	mission	C60	inreqhdr	mission	V60	Name of the mission from which data came (i.e. AM-1)
1	ProcessingStartTime	26	inreqhdr	procstartt	D8	inreqhdr	procstartt	D8	Processing start data & time for ingest of data granule
1	ProcessingEndTime	26	inreqhdr	procendti m	D8	inreqhdr	procendti m	D8	Processing end data & time for ingest of data granule
1	TimeToXfer	11	inreqhdr	time2xfer	N11.0	inreqhdr	time2xfer	Dec11.0	Time from start of transfer for 1st file in granule to time of receipt of status (success or fail) for last file in granule in seconds
1	TimeToPreprocess	16	inreqhdr	time2prepr	N16.0	inreqhdr	time2prepr	Dec16.0	Time from start of preprocessing of granule to time of completion (success or fail) of preprocessing in seconds
1	TimeToArchive	13	inreqhdr	time2archi	N13.0	inreqhdr	time2archi	Dec13.0	Date and time to archive the data (?) or amount of time to archive...must see actual data) in seconds
1	TotalDataVolume	20	inreqhdr	totaldatav	N20.1	inreqhdr	totaldatav	Dec20.1	Total data volume of granules (in Bytes) requested to be ingested, including unsuccessful, although can be 0 if the request dies immediately (Record in InReqData never created)
1	TotalFileCount	14	inreqhdr	totalfilec	N14.0	inreqhdr	totalfilec	Dec14.	Total number of files for the request

ECS Table ID	ECS (Script) Column Name	ECS (Script) Column Length	EDGRS (FoxPro) Table Name	EDGRS (FoxPro) Column Name	EDGRS (FoxPro) Column Type	EDGRS (SQL Server) Table Name	EDGRS (SQL Server) Column Name	EDGRS (SQL Server) Column Type	Description
							0		
1	TotalGranuleCount	17	inreqhdr	totgranule	N10.0	inreqhdr	totgranule	Dec10.0	Total number of granules requested
1	TotalSuccessfulGranules	23	inreqhdr	totsuccess	N15.0	inreqhdr	totsuccess	Dec15.0	Total number of granules successfully ingested
			inreqhdr	putnedgrs	D8	inreqhdr	putnedgrs	D8	Datetime record put in EDGRS db
						inreqdata	provider	V10	provider/DAAC originating this record
2	RequestID	11	inreqdata	requestid	C11	inreqdata	requestid	V11	Request identified automatically generated from the InNextAvailableID table
2	DataGranuleID	13	inreqdata	granuleid	C13	inreqdata	granuleid	V11	Data granule identifier for that requestid
2	DataType	32	inreqdata	datatype	C32	inreqdata	datatype	V13	Holds primary ESDT shortname of a ECS data type that is handled by a particular data server. (i.e. AM1 I), LandSat7
2	DataGranuleVolume	20	inreqdata	volume	N20.2	inreqdata	volume	Dec20.2	Total data volume in bytes to be ingested for a data granule in an ingest request, determined by summing the data volume for the files comprising the data granule
2	DataGranuleState	20	inreqdata	state	C20	inreqdata	state	C20	State of the data granule. Valid values: ArchErr, Archived, Cancelled, New, PreprocErr, Preprocessed, Terminated, Transferred, XferErr.
2	ProcessingStartTime	26	inreqdata	procstart	D8	inreqdata	procstart	D8	Processing start data and time for an ingest of a data granule
2	ProcessingEndTime	26	inreqdata	procend	D8	inreqdata	procend	D8	Processing end date and time for the ingest of a data granule
2	TimeToArchive	13	inreqdata	archivetim	N13.0	inreqdata	archivetim	Dec13.	Time to archive the data

ECS Table ID	ECS (Script) Column Name	ECS (Script) Column Length	EDGRS (FoxPro) Table Name	EDGRS (FoxPro) Column Name	EDGRS (FoxPro) Column Type	EDGRS (SQL Server) Table Name	EDGRS (SQL Server) Column Name	EDGRS (SQL Server) Column Type	Description
					a		0		
2	TimeToPreprocess	16	inreqdata	preprocess	N16.0	inreqdat a	preprocess	Dec16.0	Time from start of preprocessing of data granule to time of completion (success or fail) of preprocessing
2	TimeToXfer	11	inreqdata	xfertime	N11.0	inreqdat a	xfertime	Dec11.0	Time from start of transfer for 1 <sup>st</sup> file in granule to time of receipt of status (success or fail) for last file in granule
			inreqdata	putnedgrs	D8	inreqdat a	putnedgrs	D8	Datetime record put in EDGRS db
						dsmdgren	provider	V10	provider/DAAC originating this record
3	dbID	19	dsmgdran	dbid	C19	dsmgdran	dbid	V19	Unique ID identifying a database tuple.
3	ShortName	9	dsmgdran	shortname	C9(ECS) C80 (non-ECS)	dsmgdran	shortname	V80	Official reference name used in identifying the contents of the data collection
3	SizeMBECSDataGranule	20	dsmgdran	sizembecs	N20.6	dsmgdran	sizembecs	Dec20.6	Size attribute which indicates the volume of data contained in the granule (in megabytes)
3	lastUpdate	26	dsmgdran	lastupdate	D8	dsmgdran	lastupdate	D8	Time of last update
3	BeginningDateTime	26	dsmgdran	begintime	D8	dsmgdran	begintime	D8	Attribute within SDSRV that allows both SingleDateTime (TimeOfDay) and RangeDateTime (RangeBeginningDate) to be efficiently indexed and searched
3	EndingDateTime	26	dsmgdran	endtime	D8	dsmgdran	endtime	D8	Attribute within SDSRV that allows both

ECS Table ID	ECS (Script) Column Name	ECS (Script) Column Length	EDGRS (FoxPro) Table Name	EDGRS (FoxPro) Column Name	EDGRS (FoxPro) Column Type	EDGRS (SQL Server) Table Name	EDGRS (SQL Server) Column Name	EDGRS (SQL Server) Column Type	Description
					n				SingleDateTime (TimeOfDay) and RangeDateTime (RangeEndingDate) to be efficiently indexed
3	insertTime	26	dsmdgran	inserttime	D8	dsmdgran	inserttime	D8	Time of original insertion
3	LocalGranuleID	80	dsmdgran	locgranid	C100 (ECS) C240 (non-ECS)	dsmdgran	locgranid	V100	Unique identifier for locally produced granule that ECS ingests and is required to capture
			dsmdgran	putnedgrs	D8	dsmdgran	putnedgrs	D8	Datetime record put in EDGRS db
3	VersionID	5	dsmdgran	version	C4	dsmdgran	version	V10	Version of the granule
3	DeleteEffectiveDate	26	dsmdgran	deletedate	D8				Date granule was/is to be deleted
3	DeleteFromArchive	5	dsmdgran	delfromarc	C1	dsmdgran	delfromarc	C1	Delete from archive flag
			dsmdgran	duplicate	C1	dsmdgran	duplicate	C1	Duplicate flag
						dsmdgran	sortdate	C10	Converted endtime for indexing
						dsmdgran	sortobsdate	C10	Converted inserttime for indexing
						dsddrqst	provider	V10	provider/DAAC originating this record
4	RequestId	50	dsddrqst	requestid	C50	dsddrqst	requestid	V50	Request ID of distribution request that came from the Science Data Server (SDSRV)
4	OrderId	50	dsddrqst	orderid	C50	dsddrqst	orderid	V50	OrderID for the distribution request
4	State	50	dsddrqst	state	C50	dsddrqst	state	V50	Queue state of the distribution request

ECS Table ID	ECS (Script) Column Name	ECS (Script) Column Length	EDGRS (FoxPro) Table Name	EDGRS (FoxPro) Column Name	EDGRS (FoxPro) Column Type	EDGRS (SQL Server) Table Name	EDGRS (SQL Server) Column Name	EDGRS (SQL Server) Column Type	Description
									object (i.e. Pending, Active, Shipped...)
4	EcsUserId	50	dsddrqst	ecsuserid	C50	dsddrqst	ecsuserid	V50	User ID of the user initiating request
4	SizeInMB	20	dsddrqst	sizeinmb	N18.1	dsddrqst	sizeinmb	Dec20.1	Total bytes in distribution request (in bytes-not in MBs!)
4	StartTime	255	dsddrqst	starttime	D8	dsddrqst	starttime	D8	Time the distribution started
4	EndTime	255	dsddrqst	endtime	D8	dsddrqst	endtime	D8	Time distribution ended
4	NrGranules	11	dsddrqst	nrgranules	N11.0	dsddrqst	nrgranules	Dec11.0	Number of granules per request
4	NrReqFiles	11	dsddrqst	nrreqfiles	N11.0	dsddrqst	nrreqfiles	Dec11.0	Number of files in distribution request
4	NrMedia	11	dsddrqst	nrmedia	N11.0	dsddrqst	nrmedia	Dec11.0	Number of media objects (refers to tapes when >1)
5	MediaType	50	dsddrqst	mediatype	C50	dsddrqst	mediatype	V50	Type of media used for a request (i.e.8MM tape, CDRom)
5	UserProfile	50	dsddrqst	userprofil	C50	dsddrqst	userprofil	V50	Profile ID
5	FtpHost	255	dsddrqst	ftphost	C254	ftphost	ftphost	V254	Hostname to connect to for FTP push
5	FtpPushDest	255	dsddrqst	ftppushdes	C254	ftppushdest	ftppushdes	V254	Target system directory
5	FtpPullHost	255	dsddrqst	ftppullhos	C254	ftppullhost	ftppullhos	V254	Host that user will ftppull from
4	EsdtType	50	dsddrqst	esdtttype	C80	dsddrqst	esdtttype	V80	ESDT type; can include many granules but all granules associated with a request must be of same EsdtType
4	UserString	50	dsddrqst	userstring	C50	dsddrqst	userstring	V50	User string
				putnegrs	D8	dsddrqst	putnedgrs	D8	Datetime record put in EDGRS db
			dsddrqst	esdt_id	I4	dsddrqst	esdt_id	I4	Normalized datatype id
						dsddgran	provider	V10	provider/DAAC originating this record

ECS Table ID	ECS (Script) Column Name	ECS (Script) Column Length	EDGRS (FoxPro) Table Name	EDGRS (FoxPro) Column Name	EDGRS (FoxPro) Column Type	EDGRS (SQL Server) Table Name	EDGRS (SQL Server) Column Name	EDGRS (SQL Server) Column Type	Description
4	RequestId	50	dsddgran	requestid	C50	dsddgran	requestid	V50	Distribution request id
4	State		dsddgran	status	C150	dsddgran	status	V150	
6	GranuleId	150	dsddgran	granuleid	C190	dsddgran	granuleid	V190	Granule ID
6	granuleSize	20	dsddgran	granulesiz	N16.0	dsddgran	granulesiz	Dec16.1	sum of sizes of files in granule in bytes
4	EsdtType	50	dsddgran	esdtype	C80	dsddgran	esdtype	V80	DataType
4	StartTime	254	dsddgran	starttime	D8	dsddgran	starttime	D8	starttime of distribution request
			dsddgran	putnedgrs	D8	dsddgran	putnedgrs	D8	Datetime record put in EDGRS db
			dsddgran	esdt_id	I4	dsddgran	esdt_id	I4	Normalized datatype ID
						dsddgran	prodid	I4	Normalized product name
						dprs	provider	V10	provider/DAAC originating this record
7	dprId	29	dprs	dprid	C29	dprs	dprid	V29	DPR (Data processing request) identifier
7	completionDate	26	dprs	completion	D8	dprs	completion	D8	Date DPR completed
7	dprElapsed Time	20	dprs	dprelapsed	N20.8	dprs	dprelapsed	Dec20.8	Elapsed time of DPR execution
7	pgeElapsed Time	20	dprs	pgeelapsd	N20.8	dprs	pgeelapse d	Dec20.8	Elapsed time of PGE (product generation executive) execution
7	pgeBlockInputOperations	23	dprs	pgeblockin	N20.0	dprs	pgeblockin	Dec20.0	Number of block input operations

ECS Table ID	ECS (Script) Column Name	ECS (Script) Column Length	EDGRS (FoxPro) Table Name	EDGRS (FoxPro) Column Name	EDGRS (FoxPro) Column Type	EDGRS (SQL Server) Table Name	EDGRS (SQL Server) Column Name	EDGRS (SQL Server) Column Type	Description
7	pgeBlockOutputOperations	24	dprs	pgeblocko u	N20.0	dprs	pgeblocko u	Dec20. 0	Number of block output operations
7	pgeSwaps	11	dprs	pgeswaps	N11.0	dprs	pgeswaps	Dec11. 0	Number of swaps during PGE execution
7	pgeMaxMemoryUse	20	dprs	pgemaxme mo	N20.1	dprs	pgemaxme mo	Dec20. 1	Maximum memory used by PGE
7	pgeSharedMemoryU se	20	dprs	pgeshared m	N20.1	dprs	pgeshared m	Dec20. 1	Amount of shared memory used by the PGE
7	pgeCPUTime	20	dprs	pgecputim e	N20.8	dprs	pgecputim e	Dec20. 8	Amount of CPU time used by the PGE
7	systemCPUTime	20	dprs	systemcput	N20.8	dprs	systemcpu t	Dec20. 8	Amount of system time spent during execution
7	pgePageFaults	13	dprs	pgepagefa u	N13.0	dprs	pgepagefa u	Dec13. 0	Number of page faults during PGE execution
7	ExitCode		dprs	exitcode	C20	dprs	exitcode	V20	exit return code
7	platform	60	dprs	platform	C60	dprs	platform	V60	Actual science processing hardware used for this execution.
			dprs	putnedgrs	D8	dprs	putnedgrs	D8	Datetime record put in EDGRS db
					pldgshor t	provider	V10		provider/DAAC originating this record
8	granuleId	130	pldgshort	granuleid	C130	pldgshort	granuleid	V130	Identifier of a data granule
8	dataTypeId	20	pldgshort	datatypeid	C20	pldgshort	datatypeid	V20	Identifier for the data type
8	universalReference	254	pldgshort	ur	C254	pldgshort	ur	V254	Universal reference associated with this data granule
8	startTime	26	pldgshort	starttime	D8	pldgshort	starttime	D8	Start date and time for the data granule

ECS Table ID	ECS (Script) Column Name	ECS (Script) Column Length	EDGRS (FoxPro) Table Name	EDGRS (FoxPro) Column Name	EDGRS (FoxPro) Column Type	EDGRS (SQL Server) Table Name	EDGRS (SQL Server) Column Name	EDGRS (SQL Server) Column Type	Description
8	stopTime	26	pldgshort	stoptime	D8	pldgshort	stoptime	D8	Stop date and time for the data granule
8	version	11	pldgshort	version	C11	pldgshort	version	V11	Used to distinguish between two granules having same data type and start time as the result of reprocessing
8	insertTime	26	pldgshort	inserttime	D8	pldgshort	inserttime	D8	Not in 311; but time of original insertion
8	timeStamp	26	pldgshort	timestamp	D8	pldgshort	timestamp	D8	Date and time this table row was last modified.
			pldgshort	putnedgrs	D8	pldgshort	putnedgrs	D8	Datetime record put in EDGRS db
						ecacrqst	provider	V10	provider/DAAC originating this record
						ecacrqst	edrsid	I4	unique EDGRS user identifier
10	userid	12	ecacorder	userid	C12	ecacrqst	usrprofile_curr	V15	user id
9	OrderId	10	ecacrqst	orderid	C10	ecacrqst	orderid	V10	order ID
9	OrderHomeDAAC	10	ecacrqst	homedaac	C13	usrprofile_curr	homedaac	V13	where order was placed
9	RequestId	10	ecacrqst	requestid	C10 (ECS) C50 (non-ECS)	ecacrqst	requestid	V50	request ID
9	RequestProcessingDAAC	22	ecacrqst	rqstdaac	C22	ecacrqst	rqstdaac	V22	where request was processed
9	FirstName	20	ecacrqst	firstname	C20				User's first name

ECS Table ID	ECS (Script) Column Name	ECS (Script) Column Length	EDGRS (FoxPro) Table Name	EDGRS (FoxPro) Column Name	EDGRS (FoxPro) Column Type	EDGRS (SQL Server) Table Name	EDGRS (SQL Server) Column Name	EDGRS (SQL Server) Column Type	Description
9	MiddleInit	10	ecacrqst	middleinit	C10				User's middle initial
9	LastName	20	ecacrqst	lastname	C20				User's last name
9	eMailAddr	255	ecacrqst	emailaddr	C254				User's email address
9	RequestStatus	22	ecacrqst	status	C22	ecacrqst	status	V22	status of request (Pending, Staging, Shipped, etc.)
9	NumFiles	12	ecacrqst	numfiles	N8.0	ecacrqst	numfiles	Dec8.0	number of files that fills request
9	NumBytes	20	ecacrqst	numbytes	N15.0	ecacrqst	numbytes	Dec15.0	number of bytes that fill request
9	numGranule	12	ecacrqst	numgranule	N8.0	ecacrqst	numgranule	Dec8.0	number of granules that fill request
9	MediaType	20	ecacrqst	mediatype	C20	ecacrqst	mediatype	V20	media type (pull, push, 8MM, etc.)
9	ShipAddrStreet1	32	ecacrqst	ship1stree	C32	usrprofile curr	ship1stree	V32	street shipping address #1
9	ShipAddrStreet2	32	ecacrqst	ship2stree	C32	usrprofile curr	ship2stree	V32	street shipping address #2
9	ShipAddrStreet3	32	ecacrqst	ship3stree	C32	usrprofile curr	ship3stree	V32	street shipping address #3
9	ShipAddrCity	35	ecacrqst	shipcity	C35	usrprofile curr	shipcity	V35	city shipping address
9	ShipAddrState	20	ecacrqst	shipstate	C20	usrprofile curr	shipstate	V20	state shipping address
9	ShipAddrZip	15	ecacrqst	shipzip	C15	usrprofile curr	shipzip	V15	zip code of shipping address
9	ShipAddrCountry	30	ecacrqst	shipcountr	C30	usrprofile curr	shipcountr	V30	country of shipping address
9	ShipAddrPhone	22	ecacrqst	shipphone	C22	usrprofile curr	shipphone	V22	phone # at shipping address
9	ReceiveDateTime	26	ecacrqst	receivetim	D8	ecacrqst	receivetim	D8	time submitted(created) to SDSRV

ECS Table ID	ECS (Script) Column Name	ECS (Script) Column Length	EDGRS (FoxPro) Table Name	EDGRS (FoxPro) Column Name	EDGRS (FoxPro) Column Type	EDGRS (SQL Server) Table Name	EDGRS (SQL Server) Column Name	EDGRS (SQL Server) Column Type	Description
9	StartTime	26	ecacrqst	starttime	D8	ecacrqst	starttime	D8	time DDIST 1 <sup>st</sup> started processing request
9	FinishDateTime	26	ecacrqst	finishtime	D8	ecacrqst	finishtime	D8	time DDIST finished processing request
9	TimeOfLastUpdate	26	ecacrqst	lastupdate	D8	ecacrqst	lastupdate	D8	time of last update
9	ShipDateTime	26	ecacrqst	shiptime	D8	ecacrqst	shiptime	D8	date product shipped
9	FTPAddress	50	ecacrqst	ftpaddress	C50	ftp	ftpaddress	V50	req's ftp staging address
9	DestinationNode	20	ecacrqst	destnode	C20	ftp	destnode	V20	Destination node for ftp acquires
9	DestinationDirectory	20	ecacrqst	destdirect	C20	ftp	destdirect	V20	Destination directory for ftp acquires
			ecacrqst	calctime	D8	ecacrqst	calctime	I4	Minimum valid time between starttime and receivetim
			ecacrqst	useraffili	N2.0	usrprofile curr	useraffili	I4	User-provided affiliation
			ecacrqst	calcaffili	N2.0	usrprofile curr	calcaffili	I4	Calculated affiliation (from emailaddr)
9	ESDT_Id	50	ecacrqst	Esdt_Id	C80	ecacrqst	esdt_id	V80	data type from ecacrqst
4	esdttpe	50	dsddrqst	esdttpe	C80	ecacrqst dsddrqst	esdttpe	V80	data type from dsddrqst (need to get it from dsddgran)
			ecacrqst	putnedgrs	D8	ecacrqst	putnedgrs	D8	Datetime record put in EDGRS db
						ecacorder	provider	V10	provider/DAAC originating this record
						ecacorder	edgrsid	I4	unique EDGRS user identifier
10	orderId	10	ecacorder	orderid	C10	ecacorder	orderid	V10	order ID
10	orderHomeDAAC	10	ecacorder	homedaac	C13	usrprofile curr	homedaac	V13	DAAC where order placed
10	userId	12	ecacorder usrprofile	userid	C15	usrprofile curr	userid	V15	ID of user placing order
10	firstName	20	ecacrqst	firstname	C20	usrprofil	firstname	V20	user's first name

ECS Table ID	ECS (Script) Column Name	ECS (Script) Column Length	EDGRS (FoxPro) Table Name	EDGRS (FoxPro) Column Name	EDGRS (FoxPro) Column Type	EDGRS (SQL Server) Table Name	EDGRS (SQL Server) Column Name	EDGRS (SQL Server) Column Type	Description
					e_curr				
10	middleInit	10	ecacrqst	middleinit	C10	usrprofile curr	middleinit	V5	user's middle initial
10	lastName	20	ecacrqst	lastname	C20	usrprofile curr	lastname	V20	user's last name
10	eMailAddr	25	ecacrqst	emailaddr	C254	usrprofile	emailaddr	V254	user's email address
10	orderStatus	22	ecacorder	orderstatus	C22	ecacorder	orderstatus	V22	status of order
10	OrderMedia	20	ecacorder	ordermedia	C20	ecacorder	ordermedia	V20	Media type of the user's order
10	OrderGranule	12	ecacorder	ordergranu	N8.0	ecacorder	ordergranu	Dec8.0	The number of granules that fill the user's order
10	ReceiveDateTime	26	ecacorder	receivetim	D8	ecacorder	receivetim	D8	Date order received
10	StartDateTime	26	ecacorder	starttime	D8	ecacorder	starttime	D8	time DDIST 1 <sup>st</sup> started processing request
10	FinishDateTime	26	ecacorder	finishtime	D8	ecacorder	finishtime	D8	time DDIST finished processing request
10	TimeOfLastUpdate	26	ecacorder	lastupdate	D8	ecacorder	lastupdate	D8	time of last update
10	ShipDateTime	26	ecacorder	shiptime	D8	ecacorder	shiptime	D8	date product shipped
			ecacorder	putnedgrs	D8	ecacorder	putnedgrs	D8	Datetime record put in EDGRS db
						ecacorder	userid	V15	userid
						usrprofil	provider	V10	provider/DAAC originating this record

ECS Table ID	ECS (Script) Column Name	ECS (Script) Column Length	EDGRS (FoxPro) Table Name	EDGRS (FoxPro) Column Name	EDGRS (FoxPro) Column Type	EDGRS (SQL Server) Table Name	EDGRS (SQL Server) Column Name	EDGRS (SQL Server) Column Type	Description
					e_curr				
11	UserId	12	usrprofile_ecacorder	userid	C15	usrprofile_curr	userid	V50	uniquely identifies a registered user.
11	HomeDAAC	8	Usrprofile	homedaac	C10	usrprofile_curr	homedaac	V10	name of a DAAC where the Request was issued
11	FirstName	20	usrprofile	firstname	C20	usrprofile_curr_ecacorder	firstname	V20	First name
11	MiddleInit	10	usrprofile	middleinit	C10	usrprofile_curr	middleinit	V10	Middle initial
11	lastName	20	usrprofile	lastname	C20	usrprofile_curr	lastname	V20	Last name of user
11	GTWYUsrType	20	usrprofile	gtwusrtype	C20	usrprofile_curr	gtwusrtype	V20	For registered users, the gateway will retrieve their user profile and check this attribute. If is filled, it will use GTWYUsrType and a generated password to log the user into DCE (rather than the userID attribute). A DCE account for GTWYUsrType must exist with the current V0GwPwd as its password. <b>Valid Values:</b> DAACOPS DAAC Operations User, ECSDEV ECS Development User, V0CERES V0 CERES User, GUEST Guest User
11	EmailAddr	255	usrprofile	emailaddr	C254	usrprofile_curr	emailaddr	V254	User's email address

ECS Table ID	ECS (Script) Column Name	ECS (Script) Column Length	EDGRS (FoxPro) Table Name	EDGRS (FoxPro) Column Name	EDGRS (FoxPro) Column Type	EDGRS (SQL Server) Table Name	EDGRS (SQL Server) Column Name	EDGRS (SQL Server) Column Type	Description
11	InternetAffiliation	19	usrprofile	internetaf	C19	usrprofile_curr	internetaf	V19	user's internet affiliation
11	Organization	31	usrprofile	organizati	C31	usrprofile_curr	organizati	V31	User's organization
11	ProjectName	30	usrprofile	project	C30	usrprofile_curr	project	V30	user's project name
11	Affiliation	16	usrprofile	affiliatio	C16	usrprofile_curr	affiliatio	V16	user's affiliation. <b>Valid Values:</b> Gov. Research Government, Other, Univ. Research, Univ. Class Work, Commercial, Kinder.-12 Grade
11	ResearchField	64	usrprofile	researchfl	C64	usrprofile_curr	researchfl	V64	research field available in the system.
11	AccountNumber	17	usrprofile	accountnu	C17	usrprofile_curr	accountnu	V17	account number that is given by the user when they request a user profile. It becomes associated with the user profile.
11	CreationDate	26	usrprofile	createdate	D8	usrprofile_curr	createdate	D8	Userid creation date
11	MailAddrStreet1	32	usrprofile	mail1stree	C32	usrprofile_curr	mail1stree	V32	street mailing address #1
11	MailAddrStreet2	32	usrprofile	mail2stree	C32	usrprofile_curr	mail2stree	V32	street mailing address #2
11	MailAddrStreet3	32	usrprofile	mail3stree	C32	usrprofile_curr	mail3stree	V32	street mailing address #3
11	MailAddrCity	35	usrprofile	mailcity	C35	usrprofile_curr	mailcity	V35	city mailing address
11	MailAddrState	20	usrprofile	mailstate	C20	usrprofile_curr	mailstate	V20	state mailing address

ECS Table ID	ECS (Script) Column Name	ECS (Script) Column Length	EDGRS (FoxPro) Table Name	EDGRS (FoxPro) Column Name	EDGRS (FoxPro) Column Type	EDGRS (SQL Server) Table Name	EDGRS (SQL Server) Column Name	EDGRS (SQL Server) Column Type	Description
11	MailAddrZip	15	usrprofile	mailzip	C15	usrprofile_curr	mailzip	V15	zip code of mailing address
11	MailAddrCountry	30	usrprofile	mailcountr	C30	usrprofile_curr	mailcountr	V30	country of mailing address
11	MailAddrPhone	22	usrprofile	mailphone	C22	usrprofile_curr	mailphone	V22	phone # at mailing address
11	ShipAddrStreet1	32	usrprofile	ship1stree	C32	usrprofile_curr	ship1stree	V32	street shipping address #1
11	ShipAddrStreet2	32	usrprofile	ship2stree	C32	usrprofile_curr	ship2stree	V32	street shipping address #2
11	ShipAddrStreet3	32	usrprofile	ship3stree	C32	usrprofile_curr	ship3stree	V32	street shipping address #3
11	ShipAddrCity	35	usrprofile	shipcity	C35	usrprofile_curr	shipcity	V35	city shipping address
11	ShipAddrState	20	usrprofile	shipstate	C20	usrprofile_curr	shipstate	V20	state shipping address
11	ShipAddrZip	15	usrprofile	shipzip	C15	usrprofile_curr	shipzip	V15	zip code of shipping address
11	ShipAddrCountry	30	usrprofile	shipcountr	C30	usrprofile_curr	shipcountr	V30	country of shipping address
11	ShipAddrPhone	22	usrprofile	shipphone	C22	usrprofile_curr	shipphone	V22	phone # at shipping address
11	NasaUser	8	usrprofile	nasauser	C8	usrprofile_curr	nasauser	V8	This field identifies whether a user works for NASA. Valid Values: Y=Yes, N=No
			usrprofile	putnedgrs	D8	usrprofile_curr	putnedgrs	D8	Datetime record put in EDGRS db
			ecacrqst	useraffili	N2.0	usrprofile_curr	useraffili	I4	User-provided affiliation

ECS Table ID	ECS (Script) Column Name	ECS (Script) Column Length	EDGRS (FoxPro) Table Name	EDGRS (FoxPro) Column Name	EDGRS (FoxPro) Column Type	EDGRS (SQL Server) Table Name	EDGRS (SQL Server) Column Name	EDGRS (SQL Server) Column Type	Description
			ecacrqst	calcaffili	N2.0	usrprofile_curr	calcaffili	I4	Calculated affiliation (from emailaddr)
						Usrprofile_curr	firstdate	C10	first date the user ordered data
12	dprId	29	pldpreq	dprid	C30				data processing identifier
12	pgeId	17	pldpreq	pgeid	C17				Product Generation Executive identifier
12	dataStartTime	30	pldpreq	datastart	D8				The start time for the data that is input to the DPR.
12	dataStopTime	30	pldpreq	datastop	D8				The stop time for the data that is input to the DPR.
12	completionState	10	pldpreq	state	C15				A status indicator describing the current status of the DPR.
			pldpreq	putnedgrs	D8				Datetime record put in EDGRS db
13	dprid	29	pldprdata	dprid	C30				data processing identifier
7	dprid	29	pldprdata	dprid2	C28				data processing identifier
7	completionDate	29	pldprdata	completion	D8				completion date of dpr
13	granuleid	130	pldprdata	granuleid	C130				granule identifier
13	logicalId	10	pldprdata	logicalId	C10				Identifies which multiple instances of DPR/granule pairs is meant.
13	primaryType	11	pldprdata	primaryTyp	C11				Primary data type ID for this alternate input.
13	accepted	10	pldprdata	accepted	N5.0				Indicates whether the data granule, when available has passed metadata checks.
13	theOrder	10	pldprdata	order	N5.0				Order number associated with production request.
13	type	10	pldprdata	type	N5.0				Type (int, string, etc.) of the metadata parameter to be compared.
13	temporalFlag	10	pldprdata	temporalfl	N1.0				Flag indicating whether most recent

ECS Table ID	ECS (Script) Column Name	ECS (Script) Column Length	EDGRS (FoxPro) Table Name	EDGRS (FoxPro) Column Name	EDGRS (FoxPro) Column Type	EDGRS (SQL Server) Table Name	EDGRS (SQL Server) Column Name	EDGRS (SQL Server) Column Type	Description
									alternate input can be used if a primary granule of the same data type cannot be found for the production request time frame.
13	timeWait	10	pldprdata	timewait	N10.0				Associated timer.
13	ioFlag	10	pldprdata	ioflag	N1.0				ioFlag indicates whether metatdat check should be performed on input or output.
13	timeExp	10	pldprdata	timerexp	N				Indicates whether timer has expired.
13	timerStart	10	pldprdata	timerstart	N1.0				Indicates whether the timer has started.
13	numNeeded	10	pldprdata	numneeded	N10.0				The number of inputs required by the PGE.
13	waitForFlag	10	pldprdata	wait4flag	N1.0				Indicates whether DPR should be released after last timer, even if the chain is incomplete.
13	linkId	10	pldprdata	linkid	N10.0				Reserved for future use
13	minGranReq	10	pldprdata	mingranreq	N10.0				Minimum number of granules of this input type required by the PGE when forming a DPR.
			pldprdata	putnedgrs	D8				Datetime record put in EDGRS db
						subscrip	provider	V10	provider/DAAC originating this record
						subscrip	userid	C50	userid of subscription
						subscrip	scf	C20	science center
						subscrip	putnedgrs	D8	Datetime record put in EDGRS db
						subscrip	category	V20	Subscription category
						subscrip	rollup	V30	Rollup
						subscrip	user	V40	User comment field
16						ftphost	hostid	I4	Unique host identifier
5	FtpHost	10	dsddrqst	ftphost	C25	ftphost	ftphost	V254	Hostname to connect to for FTP push

ECS Table ID	ECS (Script) Column Name	ECS (Script) Column Length	EDGRS (FoxPro) Table Name	EDGRS (FoxPro) Column Name	EDGRS (FoxPro) Column Type	EDGRS (SQL Server) Table Name	EDGRS (SQL Server) Column Name	EDGRS (SQL Server) Column Type	Description
17					ftppullhos	pullid	I4		Unique ftppull address identifier
5	FtpPullHost	255	dsddrqst	ftppullhos	C254	ftppullhos	ftppullhos	V254	Host that user will ftppull from
18					ftppushdes	pushid	I4		Unique ftppush destination
5	FtpPushDest	255	dsddrqst	ftppushdes	C254	ftppushdes	ftppushdes	V254	Target system directory
19					ftp	ftpid	I4		Unique ftpaddress identifier
5	FTPAddress	50	ecacrqst	ftpaddress	C50	ftp	ftpaddress	V50	req's ftp staging address
5	DestinationNode	20	ecacrqst	destnode	C20	ftp	destnode	V20	Destination node for ftp acquires
5	DestinationDirectory	20	ecacrqst	destdirect	C20	ftp	destdirect	V20	Destination directory for ftp acquires
20			affiliation	code	N2.0				Numerical affiliation code designator
20			affiliation	affil	C20				Shortname of affiliation
20			affiliation	descrip	C50				Description of affiliation
21			actionlog	provider	C10				Provider/DAAC where data is originating from
21			actionlog	datestamp	D8				Date stamp of script execution (not ftp time)
21			actionlog	edgversion	C8				Version of EDGRS run
21			actionlog	startpull	D8				Initial date of pull window
21			actionlog	stoppull	D8				Terminal date of pull window
21			actionlog	mode	C3				operation's mode, such as OPS,TS1,TS2
21			actionlog	logtest	Memo4				output log from pull script
21			actionlog	putnedgrs	D8				Datetime record put in EDGRS db
22			err	username	C50				Name of routine trapping the error
22			err	date	D8				Date of error
22			err	time	C10				Time of error

ECS Table ID	ECS (Script) Column Name	ECS (Script) Column Length	EDGRS (FoxPro) Table Name	EDGRS (FoxPro) Column Name	EDGRS (FoxPro) Column Type	EDGRS (SQL Server) Table Name	EDGRS (SQL Server) Column Name	EDGRS (SQL Server) Column Type	Description
22			err	err_num	N6.0				Error number if known
22			err	err_line	N6.0				Line in script where error occurred
22			err	err_msg	C100				Error message
22			err	err_code	C80				Code being executed at time of error
22			err	traceback	Memo4				Traceback of calling routines at time of error
22			err	vars	Memo4				Value of all variables at time of error
23			qalog2	primarykey	D8				Datetime stamp of start of EDGRS.EXE execution
23			qalog2	msgtext	Memo4				Output of the EDGRS.EXE script
23			Qalog2	lastupdate	D8				Date of last update
24			tables	id	N3.0				table id
24			tables	edg_table	C10				EDGRS table name
24			tables	sortfield	N5.0				field used for sorting
24			tables	edgcolname	C25				EDGRS column name
24			tables	ftpcoltype	C1				input column type
24			tables	ftpcollen	N3.0				input column length
24			tables	ftpcoldec	N3.0				input column decimal length
24			tables	edgcoltype	C1				output column type
24			tables	edgcollen	N3.0				output column length
24			tables	edgcoldec	N3.0				output column decimal length
24			tables	edgconvfn	C60				EDGRS conversion function for this field
24			tables	ftpcolname	C25				output column name
25				dpgranule	DBID	C12	dpgranule	V10	provider
25	DLGranuleAccess.dbf	10					dbid	V20	Granule id

ECS Table ID	ECS (Script) Column Name	ECS (Script) Column Length	EDGRS (FoxPro) Table Name	EDGRS (FoxPro) Column Name	EDGRS (FoxPro) Column Type	EDGRS (SQL Server) Table Name	EDGRS (SQL Server) Column Name	EDGRS (SQL Server) Column Type	Description
	d				le				
25	DIGranuleAccess.accessType	10	dpgranule	accesstype	C13	dpgranule	accesstype	V20	Type of access (FTP, http)
25	DIGranuleAccess.age	10	dpgranule	age	N10.0	dpgranule	age	I4	Age of granule in days(?)
25	DIGranuleAccess.file size	16	dpgranule	filesize	N16.0	dpgranule	filesize	Dec18.0	Size of the file in Megabytes
25	DIGranuleAccess.file type	10	dpgranule	filetype	C11	dpgranule	filetype	V20	File type (BROWSE, METADATA, SCIENCE)
25	DIGranuleAccess.accessTime	26	dpgranule	accesstime	D8	dpgranule	accesstime	D8	Date and time of access
25	DIGranuleAccess.ecs Id	26	dpgranule	ecsid	C26	dpgranule	ecsid	V26	ECS ID
25	DIGranuleAccess.transferTime	26	dpgranule	xfertime	C26	dpgranule	xfertime	V26	Time to transfer
25	DIGranuleAccess.specialActionFlag	26	dpgranule	actionflag	C26	dpgranule	actionflag	V26	Special action flag
25	DIGranuleAccess.ip Address	26	dpgranule	ipaddress	C26	dpgranule	ipaddress	V26	IP address
25	DIGranuleAccess.domainName	26	dpgranule	domainname	C26	dpgranule	domainname	V26	Domain name
25	DIFile.fileName	80	dpgranule	filename	C240	dpgranule	filename	V240	File name
25	DIFile.granuleId	10	dpgranule	granuleID 2	C12				granuleId=dbid
25	DIGranules.shortname	10	dpgranule	shortname	C12	dpgranule	shortname	V20	Datatype shortname
25	?		dpgranule	Filesize2	N16.0				WHERE DOES THIS COME

ECS Table ID	ECS (Script) Column Name	ECS (Script) Column Length	EDGRS (FoxPro) Table Name	EDGRS (FoxPro) Column Name	EDGRS (FoxPro) Column Type	EDGRS (SQL Server) Table Name	EDGRS (SQL Server) Column Name	EDGRS (SQL Server) Column Type	Description
									FROM???????????
26					users	edgrsid	I4		Derived user ID
26					users	createdate	D8		Date this record was created
26					users	firstname	V20		User's firstname
26					users	middleinit	V10		User's middle init
26					users	lastname	V20		User's lastname
26					users	emailaddr	V254		Users' emailaddr
26					users	Ship1stree	V32		Street part1 of shipping address
26					users	Ship2stree	V32		Street part2 of shipping address
26					users	Ship3stree	V32		Street part3 of shipping address
26					users	shipcity	V35		City of shipping address
26					users	shipstate	V20		State of shipping address
26					users	shipzip	V15		Zip code of shipping address
26					users	shipcountr	V30		Country of shipping address
26					users	shiphphone	V22		Telephone number at shipping address
26					users	useraffili	I4		User-supplied affiliation
26					users	calcaffili	I4		Calculated affiliation based on emailaddr
26					users	putnedgrs	D8		Date record was put into EDGRS
26					users	firstdate	C10		First date this user ordered data

Upper case is FoxPro function, Lower case is User-defined function

## Appendix C Processing Control Structures

The following table documents the SQL Server control environment.

Function	File Name	Procedure Name	Source Table	Destination Table	Constraining Fields Used	Other Fields Used	Comments
<b><i>Ingest Subsystem</i></b>							
DTS	10daysinsert_inreqhdrtemp.sql	NA	E0inreqhdr, g0inreqhdr, l0inreqhdr, n0inreqhdr	Inreqhdrtemp (SQL Server-SS)	TODAY-10>=procstartt<=TODAY; not empty(procstartt)	Provider, *	Import data from FoxPro to Sql Server
DTS	10daysinsert_inreqdatatemp.sql	NA	E0inreqhdr, g0inreqhdr, l0inreqhdr, n0inreqhdr	Inreqdatatemp (SQL Server-SS)	TODAY-10>=procstartt<=TODAY; not empty(procstart)	Provider, *	Import data from FoxPro to Sql Server
Normalize	Update_Inreqhdr_curr.sql	Pro_update_inreqhdr	Inreqhdr, inreqhdrtemp	Inreqhdr	Update, insert based on provider, requestid	Insert all	Runs 10 day insert and then normalizes
Normalize	Update_Inreqdata_curr.sql	Pro_update_inreqdata	Inreqdata, inreqdatatemp	Inreqdata	Update, insert based on provider, requestid, granuleid	Insert all	Runs 10 day insert and then normalizes
Report	Sps0_ingest_prod_curr_new.sql	View_rp_ingest Pro_rp_ingest_instrument_new	Inreqdata, datatype	Rp_ingest_instrument_datatype	State-'Successful' or 'Archived', Join on datatype	Datatype: Edginstrum, mission, level Inreqdata: datatype, provider, count, volume, extdatatype, procend	Creates insertime based on procend
Report	Sps0_ingest_prod.sql	Pro_rp_ingest_summary	Rp_ingest_instrument_Datatype	Rp_ingest_summary	None	Counts, smb	Group by provider, instrument, insertime

<b>Function</b>	<b>File Name</b>	<b>Procedure Name</b>	<b>Source Table</b>	<b>Destination Table</b>	<b>Constraining Fields Used</b>	<b>Other Fields Used</b>	<b>Comments</b>
Graph	Graph_ingest_request.sql	View_graph_Ingest_request	Rp_ingest_instrument_datatype		provider	Week (based on insertime), cnt	

Expand Table - TBS

## **Appendix D      Detailed Description of Correction for Orders Spanning Days**

Errors in order ID counts occur when dtu reports which cover more than one day and contain order IDs that span multi-days are generated from the dtu summary tables. This occurs because the tables contain order ID counts grouped by day. Since the number of order IDs that cause these ‘order-spanning’ errors are relatively sparse, one can effectively correct for these errors in the following way.

1. Determine which order IDs span more than one day in the entire dist base table.
2. Get all the detail records for those order IDs.
3. Store that information in the orderspandetail table.

Then, whenever a report that contains order ID counts and covers a particular day range is generated

4. Get the correction in the order ID count by getting the difference between the order ID count obtained by including ‘day’ in the group by clause and the order ID count obtained by not including ‘day’ in the group by clause from the data in the orderspandetail table for the range specified in the report.
5. Subtract this correction from the order ID count generated by the report to get the corrected value.

**Note:** The correction can be done dynamically while generating the report.

An example of the application of this correction follows.

### **1. Query without correction for order-spanning**

```
select c.provider, sum(c.orders) orders, sum(c.requests) requests,
       sum(c.cnt) cnt, sum(c.files) files, sum(c.smb)/1048576 megab
  from rp_dtu_summary c
 where c.insertime>='2004-06-01'
 GROUP BY c.provider
```

### **2. Query with correction for order-spanning**

```
-- Apply corrections to orders and requests for desired query
SELECT e.provider as [Metrics Provider],
CASE when d.provider IS NOT NULL and d.orders > 0 then e.orders - d.orders
     else e.orders END [#Orders],
CASE when d.provider IS NOT NULL and d.requests > 0 then e.requests - d.requests
     else e.requests END [#Requests],
e.cnt [#Granules], e.files [#Files],
e.megab [#Megabytes]
FROM
-- Desired query
(select c.provider, sum(c.orders) orders, sum(c.requests) requests,
       sum(c.cnt) cnt, sum(c.files) files, sum(c.smb)/1048576 megab
  from rp_dtu_summary c
```

```

where c.insertime>='2004-06-01'
GROUP BY c.provider) as e
left outer join
-- Get corrections from differences between incorrect values and correct values
(select a.provider, a.orders - b.orders orders,
a.requests - b.requests requests
from
(select z.provider, sum(z.orders) orders, sum(z.requests) requests
from
-- Get incorrect values based on orderspandetail
(select provider, insertime,
count(distinct orderid) orders, count(distinct requestid) requests
from view_orderspandetail
WHERE insertime>='2004-06-01'
group by provider, insertime) AS z
group by z.provider) AS a
inner join
-- Get correct values based on orderspandetail
(select provider,
count(distinct orderid) orders, count(distinct requestid) requests
from view_orderspandetail
WHERE insertime>='2004-06-01'
group by provider) AS b
on a.provider = b.provider) AS d
on e.provider = d.provider
ORDER BY e.provider

```

### 3. Results

DAAC	#Ords	#Reqs	#Grans	#Files	#MBs
ASF	330	368	1951	1951	119489.785156
EDC	6821	63642	264466	1229799	25515164.562468
GHRC	56	113	9771	9771	339989.191961
GSFC	5173	131571	362720	846778	27859544.821189
GSFCV0	2620	420756	420756	420756	4529663.376953
JPL	204	300	110431	110431	1025060.083056
LaRC	634	149252	187474	404057	12348511.638916
Latis	386	637	275041	275041	2935528.695388
NSIDC	1544	8757	43695	90725	3107281.248575
ORNL	103	103	335	0	93332.941247

Table 1 – no correction for order-spanning

DAAC	#Ords	#Reqs	#Grans	#Files	#MBs
ASF	322	368	1951	1951	119489.785156
EDC	6762	63642	264466	1229799	25515164.562468
GHRC	56	113	9771	9771	339989.191961
GSFC	4996	131571	362720	846778	27859544.821189
GSFCV0	2620	420756	420756	420756	4529663.376953
JPL	204	300	110431	110431	1025060.083056
LaRC	629	149252	187474	404057	12348511.638916
Latis	386	637	275041	275041	2935528.695388
NSIDC	1540	8757	43695	90725	3107281.248575
ORNL	103	103	335	0	93332.941247

Table 2 – correction for order-spanning

#### Notes:

1. The corrected values are either lower or the same as the actual values
2. Only the count of order IDs (#Ords) is affected by the correction.